American Journal of Artificial Intelligence
2024, Vol. 8, No. 2, pp. 81-113
https://doi.org/10.11648/j.ajai.20240802.17

Research Article

Infobody Structures for Logical Artificial Intelligence with Database Implementation

Yuhu Che*

Retired Independent Researcher, Marietta, Georgia, The United States

Abstract

The purpose of this paper is to explore the applications of infobody concepts, infobody structures and infobody charts to Artificial Intelligence (AI), specifically, Logical Artificial Intelligence (LAI). It is also trying to explore a new way to resolve some logical issues in current Artificial Intelligence studies with ChatGPT such as answering reasoning questions in family relations. For this purpose, detailed family relations are discussed based on relation theory. Some new concepts such as primary relations, reversed relations and derived relations for family relations are introduced. Also, a relational database is introduced to implement these family relations and the relationships between these family relations, and make them calculatable with SQL. Each SQL query becomes an infobody processor and together with the input and output infobodies compose a unit infobody structure. Multiple unit structures compose an answer structure to answer a specific question in family relations. A specific unit structure can join multiple answer structures to answer multiple questions. A processor with related input infobodies contains all detailed information for reasoning to a specific output infobody and therefore an answer structure can answer a specific reasoning (logical) question. Each answer structure can be presented in an infobody chart which is a visualization of an infobody model. An infobody model can be implemented in another relational database that can be queried by SQL as well. Suppose all academic areas are implemented in knowledge structures with infobody models in clouds, and all commonsense areas such as family relations are implemented in thinking structures with infobody models in clouds, then, any logical AI app should be able to query some of them to answer any logical questions. Also, it is possible to make those IB models for LAI available for all kinds of robots to simulate creative thinking.

Keywords

Brain Image, Information (New Definition), Infobody, Internal Infobody, External Infobody, Infobody Structure, Infobody Chart, Infobody Model, Thinking Structure, Unit Structure, Answer Structure

1. Introduction

As Roivainen indicated in [1], ChatGPT is known to fail tasks that require real humanlike reasoning or an understanding of the physical and social world. ChatGPT easily fails at obvious riddles, such as "What is the first name of the father of Sebastian's children?" (ChatGPT on March 21, 2023: I'm sorry, I cannot answer this question as I do not have enough context to identify which Sebastian you are referring to.) It seems that ChatGPT fails to reason logically and tries to rely on its vast database of "Sebastian" facts mentioned in online texts.

*Corresponding author: yuhuche@yahoo.com (Yuhu Che)

Received: 15 October 2024; **Accepted:** 4 December 2024; **Published:** 23 December 2024

The algorithms for current AI are based on Neuronal Network and Backpropagation (Learning and Deep Learning) to train a model such as a Language Model for ChatGPT and other AI apps for estimating the parameters in the model with a huge amount of data (Big Data) and then use the trained model to answer the questions asked by the users, or help in decision-making, see Al-Masri [2]. These approaches are similar to traditional statistical analysis such as least square regression and maximum likelihood estimate which are also based on a lot of data. But there is no logical reasoning in the model and so there is no way to answer any logical questions. [3-6] talk about some concepts and methods for Logical AI (LAI) but those methods are not easy to implement in mathematical models and databases.

Infobody Model is different from current AI based on big data. Che [7] presented a new concept - Infobody and other related concepts and a generic Infobody Model in terms of Graph Theory for describing and analyzing any kind of information systems including computer systems, management systems, knowledge structures and human thinking structures, and etc. An infobody model is composed of input infobodies, processors and output infobodies. The input infobodies and the processors contains all detailed information about the reasons and mechanisms why and how the output infobodies can be created. Specifically, the infobody models for knowledge structures and thinking structures can be applied to Logical Artificial Intelligence (LAI) which is the first attempt in this paper. Relatively the current AI based on data can be regarded as Data Artificial Intelligence (DAI). LAI and DAI are not conflicting but complementary. In fact, DAI can be applied to each processor in an infobody model and the whole infobody model can describe the overall LAI structure.

2. Internal Infobodies and Human Intelligence

When a baby was born, she started to feel the surrounding world and remember some feelings as good and some feelings as bad. Those feelings become information stored in her memory. For example, when she drinks milk, she has a good feeling and keeps this feeling in her memory. If for some reason she drinks tea and she feels bad, she also remembers this bad feeling. Next time when she tastes milk and tea, she would select the milk and would not select the tea. In fact, she starts to think and compare and make decisions. All these activities happen in her mind (brain) and use her intelligence. In the whole process she never feels any neurons and nerves. In fact, no one can tell any neurons and nerves when thinking using his/her own intelligence. “Neurons” and “Nerves” are all medical terms from neurosurgeons from outside of a patient, but they work inside of human brain for memory and thinking as mind.

Hansotia [8] did a detailed review and said in the conclusion: "We conclude that the mind is a complex function, an algebraic sum of many functions of the brain. Consciousness and cognition are the essential elements without which the other qualities of mind do not seem to register. All this evidence brings us to the conclusion that the brain is the organ of the mind and that the quality of the mind depends on the quality of the brain".

A human being cannot feel those physical organs inside of his/her body but his/her memory and thinking (mind) work by these physical organs and generate those feelings. We call those feelings brain (mind) images that the human being can "see", can "hear", can "smell", can "taste" from his/her brain (mind).

The above descriptions are related to the definition of Information and so related to the definition of Infobody. Currently the definition of information is not very accurate. Merriam Webster Dictionary [9] provides a popular definition of information as below.

a. (1): knowledge obtained from investigation, study, or instruction
   (2): intelligence, news
   (3): facts, data
b. the attribute inherent in and communicated by one of two or more alternative sequences or arrangements of something (such as nucleotides in DNA or binary digits in a computer program) that produce specific effects
c. (1): a signal or character (as in a communication system or computer) representing data
   (2): something (such as a message, experimental data, or a picture) which justifies change in a construct (such as a plan or theory) that represents physical or mental experience or another construct
d. a quantitative measure of the content of information
   specifically: a numerical quantity that measures the uncertainty in the outcome of an experiment to be performed

Based on this popular definition of information, Che defined a new concept - Infobody in [7] as: Any physical body that contains information is called an Infobody. In this definition, we cannot understand what exactly the information is and how the infobody contains the information. So, the definition of infobody in [7] is not accurate enough and it is actually caused by the popular inaccurate definition of information such as in Webster Dictionary.

To improve the definition of infobody, we need to improve the definition of information first. The fact that when a baby is born and starts to feel the surrounding world can help us to define information accurately. Many brain images are created in the baby’s brain (mind) when the baby sees something by eyes, hears some sound by ears, smells something by noses, tastes something by tongue and feels something by skin. In fact, the brain images are created in every person’s brain (mind). We now define *information* as the collection of all the brain images for a specific person. Each brain image is called a piece of information for the specific person.

Based on this definition of information, we then can define

infobody as any physical body that can create or recall brain images (information) for a person. For example, a car is an infobody, a book is an infobody, the text in a book is also an infobody, each word in a text is also an infobody, each letter in a word is also an infobody.

All brain images (information) themselves are also infobodies. Those infobodies in human brains (minds) are called internal infobodies and all those infobodies that are outside of human brains (minds) are called external infobodies. We prefer to use the term "brain images" than "mind images" for reminding the physical bodies for the images.

Note here, when a neurosurgeon is doing a surgery for a patient's brain, the doctor sees the brain of the patient, this brain is an external infobody to the doctor. The patient cannot see his/her own brain. The only thing he/she can "see" - not by eyes but by internal nerves - are brain images in his/her mind (ignoring the anaesthetization for surgery).

The reality infobodies and translative infobodies (language and text) defined in [7] are all external infobodies.

Another thing is important to be mentioned here is that a brain can translate brain images into language and text in the brain (mind). But usually, we consider language as in speaking by mouth and text as in writing or printing by hand or printer on a piece of paper. So, the internal language and text in mind are just considered as brain images (information).

Internal infobodies and external infobodies are totally different. The internal infobodies are for each specific person and can be felt by that person only in his/her mind. All other infobodies are external infobodies.

The neural network of a person is actually an external infobody structure to a neurosurgeon. That specific person can never really see his or her neural network in his or her brain and other organs. What the person can “see” or feel are only the internal infobodies which are the brain images stored in his or her brain (mind). Human intelligence is all about internal infobody structures but not external infobodies. Therefore, artificial intelligence should simulate internal infobody structures but not external infobody structures.

An infobody processor (generator) is also a physical body that processes some infobodies and generates some new infobodies. The processed infobodies are called input infobodies and the generated infobodies are called output infobodies.

The output infobodies from a processor (generator) can be input infobodies for other processors (generators) to generate other output infobodies. All those input infobodies, processors (generators) and output infobodies compose an infobody structure. The mathematical model to present an infobody structure in terms of Graph Theory is called an Infobody Model which can be implemented in a relational database.

Current AI studies are more on external infobody structures such as neural network but almost no studies on internal infobody structures.

We call the internal infobody structures in human mind *Thinking Structures* which are really human intelligence. When we use our intelligence to think about any problems, no one can tell which neurons are working and which nerves are working and how they are working. The only things we can tell in our mind are our memories and try to answer some questions and get some conclusions which are all internal infobodies.

A lot of human intelligence (thinking structures) have been translated into books and articles by authors. If we translate books and articles into infobody structures, we call them *Knowledge Structures*. Simulating knowledge structures should also be the tasks of artificial intelligence.

Infobody structures can be applied to almost any area such as Computer Systems, Management Systems, Human Intelligence (Thinking Structure), Knowledge Structures, and etc. For more details about infobody and infobody structures, see [7] (Chapter 1: Infobody Overview).

An infobody structure is composed of at least one processor (generator), at least one input infobody to the processor and at least one output infobody generated by the generator (processor). The key is the processor. For thinking structure, the processor is a person’s brain (mind). An infobody model is a mathematical model (in terms of Graph Theory) to describe the infobody structure, such as thinking structure, and implemented with some computer tools such as a relational database. For thinking structure, this model should simulate the thinking process in human mind (brain).

For example, the Father-Children question mentioned in Section 1 (Introduction) would cause the human brain to think about the relations between people such as who is whose father and who are whose children. Let’s put the question here again:

What is the first name of the father of Sebastian’s children? (1)

The whole thinking process of a human being for this question would be like this: The children are Sebastian’s children, then Sebastian is the children’s father – Sebastian is a male first name by the commonsense, and so the first name of the children’s father is Sebastian. That is the answer to question (1). This thinking process (structure) can be demonstrated in a simple infobody chart ([7] Chapter2: Infobody Charting) as showing in Figure 1.

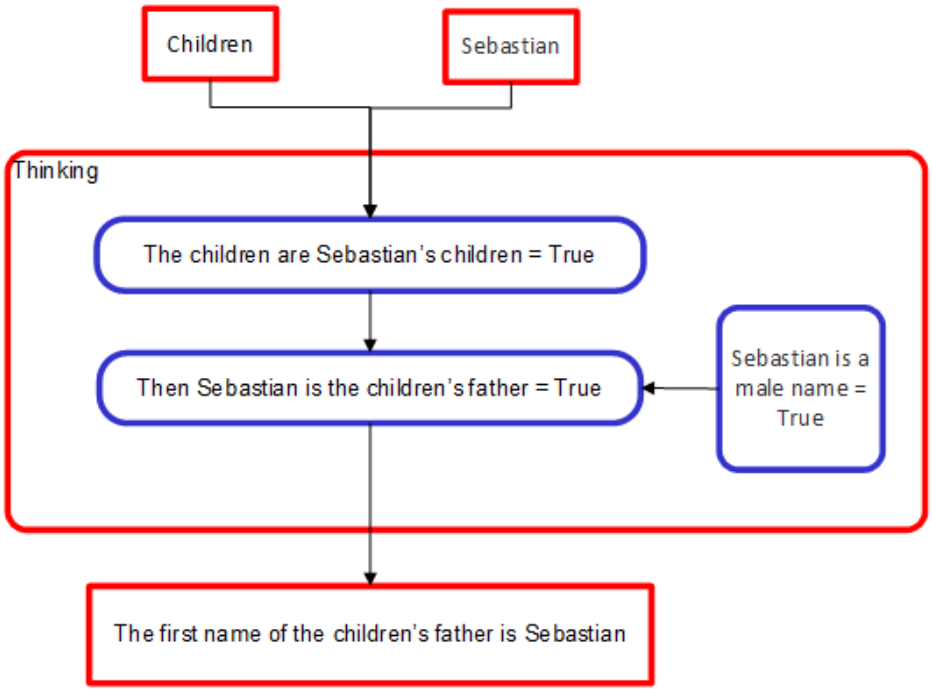

Figure 1. *Human Thinking Structure for Question (1).*

This answer is correct, but it is by human thinking. The purpose of the Artificial Intelligence is to use computer tools to simulate this thinking process and give the correct answer. Furthermore, people may ask many other similar or more complicated questions to ChatGPT or other AI apps. This means we need to think some other more complicated ways to implement this thinking process with computer tools. As a common methodology, computer implementation is often based on some kind of mathematical models. For this specific kind of models, it is related to a mathematical theory called Relation Theory.

3. Static Relations Between Infobodies

The general definition of relation in Relation Theory can be found in many books and online articles such as [10]. We will use a simplified definition and will modify it for our purpose.

Suppose we have a base set, usually called universe, denoted as U, and we will talk about the relations between the elements in U. Since no time changes, these relations are static.

A relation R in U is defined as a subset of the Cartesian product $U \times U$, or in Set Theory expression:

$$R \subseteq U \times U \tag{2}$$

For example, assume we consider a four people family with a father (denoted as p_1), a mother (denoted as p_2), a son (denoted as p_3), and a daughter (denoted as p_4). These four people compose the universe U. In set theory expression, we have:

$$U = \{p_1, p_2, p_3, p_4\} \tag{3}$$

This universe set can also be listed in a table as shown in Table 1.

Table 1. *Universe (U) Party Table.*

ID	Party	Role	Description
1	p_1	Father	Person
2	p_2	Mother	Person
3	p_3	Son	Person
4	p_4	Daughter	Person

Note, we use term "Party" for a person because we will extend individual person to groups of people. The term "Role" indicates the position of a person in the family. A role seems the same as a relation but we need to understand it is not a relation. A role roughly indicates the position of a person in the family. A relation exactly indicates the ordered pairs of people for that relation.

Since we focus on relations here, we will use infobody expression [Relation] to indicate a relation as an infobody and distinguish it from a role. Suppose we want to define the $[Father]$ relation in U, it should be:

$$[Father] = \{(p_1, p_3), (p_1, p_4)\} \tag{4}$$

Formular (4) indicates two ordered pairs of people that are in relation [Father]: (p_1, p_3) and (p_1, p_4). (p_1, p_3) means p_1 is p_3's father. (p_1, p_4) means p_1 is p_4's father as well.

Similarly, we have following relations:

$$[Mother] = \{(p_2, p_3), (p_2, p_4)\} \quad (5)$$

$$[Son] = \{(p_3, p_1), (p_3, p_2)\} \quad (6)$$

$$[Daughter] = \{(p_4, p_1), (p_4, p_2)\} \quad (7)$$

It is important to understand the difference between a role and the relation with the same term such as role "Father" and relation [Father]. The role "Father" means a person plays a "Father" role in the family. The relation [Father] means all the party pairs that the first party is the father of the second party. Usually, we call the first party *FROM Party* and the second party *TO Party*. For example, (p_1, p_3) is in [Father] relation meaning p_1 is p_3's father and p_1 is the From Party and p_3 is the To Party. A relation may contain more than one pair of parties. For example, p_1 is p_4's father as well.

You may notice that we are missing some relations in this four people family like:

$$[Parents],\ [Parent],\ [Children] \text{ and } [Child] \quad (8)$$

Those are related to groups of people and synonyms. Here we need to distinguish group and synonym. "Parents" is a group meaning both father and mother. "Parent" is a synonym for either father or mother but not both. The same difference also applies to "Children" and "Child". In terms of family relations, a group does create a new relation, but a synonym does not create a new relation.

Before discussing the new relations composed by groups, we need to modify definition (2) to include groups for relations.

A group of people is a subset of the universe U and so we will need to redefine a relation in the power set (see [11]) of the universe U, denoted as $P(U)$.

A relation R in the power set of U, $P(U)$, is defined as a subset of the Cartesian product $P(U) \times P(U)$, or, in Set Theory expression:

$$R \subseteq P(U) \times P(U) \quad (9)$$

Adding the two groups "Parents" and "Children" into Table 1, it becomes Table 2 as shown below.

Table 2. *Small Family Party Table with Groups.*

ID	Party	Role	Description
1	p_1	Father	Person
2	p_2	Mother	Person
3	p_3	Son	Person
4	p_4	Daughter	Person
5	p_5	Parents	Group
6	p_6	Children	Group

Note, Table 2 is not the power set of U, because it does not contain all subsets of U. For example, subset $\{Son, Mother, Daughter\}$ is not included. But it does include all possible roles in the four people family (Table 1). We can simply call it a small family set. From this table we can see that the number of roles depends on the language (English). In some other languages the small family set may have more roles.

As mentioned before, "Parent" is not really a group but a synonym to either father or mother but not both. Therefore, it is not listed in the party table (Table 2). For the same reason, "Child" is not listed in the table either.

With the newly added parties, the four existing relations (4) – (7) need to be modified:

$$[Father] = \{(p_1, p_3), (p_1, p_4), (p_1, p_6)\} \quad (10)$$

$$[Mother] = \{(p_2, p_3), (p_2, p_4), (p_2, p_6)\} \quad (11)$$

$$[Son] = \{(p_3, p_1), (p_3, p_2), (p_3, p_5)\} \quad (12)$$

$$[Daughter] = \{(p_4, p_1), (p_4, p_2), (p_4, p_5)\} \quad (13)$$

Some new relations can be created with groups:

$$[Parents] = \{(p_5, p_3), (p_5, p_4), (p_5, p_6)\} \quad (14)$$

$$[Children] = \{(p_6, p_1), (p_6, p_2), (p_6, p_5)\} \quad (15)$$

For synonyms, they cannot compose new relations, but in some cases, they can merge two (or even more) relations. For example, "Parent" means either "Father" or "Mother" but not both, then the relation [Parent] should be the union of relation [Father] and [Mother]:

$$[Parent] = [Father] \cup [Mother] =$$

$$\{(p_1, p_3), (p_1, p_4), (p_1, p_6), (p_2, p_3), (p_2, p_4), (p_2, p_6)\} \quad (16)$$

Here [Father] and [Mother] are called *key relations* of the synonym [Parent]. Most synonyms have only one key relation. For example, [Mom] is for [Mother] only. [Mommy] is for [Mother] only as well. We can see that one key relation may have more than one synonym but most synonyms have only one key relation for each. In only few cases, a synonym may have more than one key relations. Table 9 (in Section 7: Da-

tabase Implementation) lists most synonyms with key relations for database implementation.

Another synonym relation [Child] is listed below:

$$[Child] = [Son] \cup [Daughter] =$$

$$\{(p_3, p_1), (p_3, p_2), (p_3, p_5), (p_4, p_1), (p_4, p_2), (p_4, p_5)\} \quad (17)$$

4. Primary Family Relations

The discussions above are to introduce the basic mathematical concepts in Family Relations based on a small family in Table 2. Now we are going to use a bigger family in Table 3 as an example to show more details in Logical AI. Note, Table 3 is not an extension to Table 2 and so the party numbers are totally different.

Table 3. *Bigger Family Party Table.*

Party	Role	Description
p1	Husband	Person
p2	Wife	Person
p3	Son1	Person
p4	Son2	Person
p5	Sons	Group {p3, p4}
p6	Daughter1	Person
p7	Daughter2	Person
p8	Daughters	Group {p6, p7}
p9	Parents	Group {p1, p2}
p10	Children	Group {p3, p4, p6, p7}
p11	Grandfather1	Person - Husband's father
p12	Grandfather2	Person - Wife's father
p13	Grandfathers	Group {p11, p12}
p14	Grandmother1	Person - Husband's mother
p15	Grandmother2	Person - Wife's mother
p16	Grandmothers	Group {p14, p15}
p17	Grandparents	Group {p11, p12, p14, p15}
p18	Grandparents1	Group - Husband's parents {p11, p14}
p19	Grandparents2	Group - Wife's parents {p12, p15}

Similar to formulas (10) – (17), we list all primary relations in Table 4 based on the parties listed in Table 3. A *primary relation* means a relation for which all party pairs are composed with the parties listed in a party table like Table 3.

Table 4. *Primary Family Relations Table.*

rID	Relation	Party Pairs
1	[Husband]	(p11, p14), (p1, p2), (p12, p15)
2	[Wife]	(p14, p11), (p2, p1), (p15, p12)
3	[Father]	(p1, p4), (p1, p5), (p1, p6), (p11, p1), (p12, p2), (p1, p10), (p1, p7), (p1, p8), (p1, p3)
4	[Mother]	(p2, p4), (p2, p5), (p2, p7), (p15, p2), (p2, p10), (p14, p1), (p2, p8), (p2, p6), (p2, p3)
5	[Son]	(p3, p1), (p4, p1), (p1, p14), (p1, p18), (p1, p11), (p4, p2), (p3, p2)
6	[Sons]	(p5, p1), (p5, p2)
7	[Daughter]	(p6, p1), (p7, p1), (p2, p12), (p2, p19), (p2, p15), (p7, p2), (p6, p2)
8	[Daughters]	(p8, p1), (p8, p2)
9	[Parent]	(p12, p2), (p1, p4), (p1, p5), (p1, p6), (p1, p7), (p2, p7), (p1, p8), (p14, p1), (p2, p10), (p2, p4), (p2, p5), (p2, p6), (p11, p1), (p15, p2), (p1, p10), (p2, p8), (p2, p3), (p1, p3)
10	[Parents]	(p9, p3), (p9, p5), (p9, p7), (p18, p1), (p19, p2), (p9, p10), (p9, p8), (p9, p6), (p9, p4)
11	[Child]	(p3, p1), (p4, p1), (p6, p2), (p7, p1), (p1, p11), (p2, p15), (p2, p19), (p1, p18), (p2, p12), (p1, p14), (p7, p2), (p6, p1), (p4, p2), (p3, p2)
12	[Children]	(p10, p1), (p5, p1), (p8, p1), (p5, p9), (p8, p9), (p8, p2), (p10, p9), (p5, p2), (p10, p2)
13	[Brother]	(p3, p4), (p3, p7), (p4, p6), (p4, p7), (p4, p8), (p3, p8), (p4, p3), (p3, p6)
14	[Brothers]	(p5, p6), (p5, p8), (p5, p7)

rID	Relation	Party Pairs
15	[Sister]	(p6, p3), (p6, p5), (p7, p3), (p7, p5), (p7, p6), (p7, p4), (p6, p7), (p6, p4)
16	[Sisters]	(p8, p3), (p8, p5), (p8, p4)
18	[Grandfathers]	(p13, p3), (p13, p5), (p13, p7), (p13, p10), (p13, p8), (p13, p6), (p13, p4)
20	[Grandmothers]	(p16, p3), (p16, p5), (p16, p7), (p16, p10), (p16, p8), (p16, p6), (p16, p4)
22	[Grandparents]	(p17, p3), (p17, p5), (p17, p7), (p17, p10), (p17, p8), (p17, p6), (p17, p4)

5. Relationships Between Relations

We use the term "Relationship" to discuss the relationships between relations to avoid unclear phrases like "relations between relations".

In Figure 1, the human thinking process contains two steps:

<The children are Sebastian's children = True> (18)

<Then Sebastian is the children's father = True> (19)

The first one contains a family relation [Children] and the second one contains another family relation [Father]. They imply a relationship between the two family relations [Children] and [Father]. We call the relationship between the two family relations [Children] and [Father] *reverse*. Or we say [Father] is a reverse relation of [Children] and [Children] is a reverse relation of [Father].

Let's use the Set Theory language to define it formally.

Suppose we have two relations R_1 and R_2, if there are two parties p_i and p_j that $(p_i, p_j) \in R_1$ and $(p_j, p_i) \in R_2$, then we call R_1 and R_2 are *reversible* by (p_i, p_j) and (p_j, p_i), and $R_1$is called a *reverse of* R_2 and R_2 is called a *reverse of* R_1. The party pair (p_i, p_j) is called a *reverser* in R_1 and (p_j, p_i) is called a *reverser* in R_2. Also, we denote $R_2 = \mathrm{R}(R_1)$ and $R_1 = \mathrm{R}(R_2)$. We also denote $(p_i, p_j) = \mathrm{R}(p_j, p_i)$ for the party pair. The set of all reversers in R_1 is called a *reverser set* in R_1, denoted as $\sigma(R_1)$, and the set of all reversers in R_2 is called a *reverser set* in R_2, denoted as $\sigma(R_2)$. If $\sigma(R_1) = R_1$ and $\sigma(R_2) = R_2$ then we call R_1 and R_2 are *fully reversible*.

For example, from Table 4, the family relations [Father] and [Children] are reversible by (p_1, p_{10}). R [Father] =[Children] and R [Children] = [Father]. The reverser set

$$\sigma[Father] = \{ (p_1, p_{10}) \} \quad (20)$$

$$\sigma[Children] = \{ (p_{10}, p_1) \} \quad (21)$$

Note, a relation may have more than one reverses. For example, in Table 4, [Father] and [Son] are reversible by (p_1, p_3) and (p_3, p_1), and [Father] and [Daughter] are reversible by (p_1, p_6) and (p_6, p_1).

If for a relation R, $\mathrm{R}(R) = R$ by any $(p_i, p_j) \in R$, then we call R is *mutual*. For a mutual relation R we have:

$$R = \mathrm{R}(R) = \sigma(R) \quad (22)$$

For example, relation [Cousin] is a mutual relation because if p_i is p_j's cousin then p_j must be p_i's cousin. Note, relation [Brother] is not a mutual relation because a son is a daughter's brother but the daughter is not the son's brother.

For primary relations in Table 4, it is easy to find a reverse of a relation. For example, [Husband] and [wife] are reversible by (p_1, p_2) and (p_2, p_1). Actually, they are fully reversible because $\sigma[Husband] = [Husband]$ and $\sigma([Wife]) = [Wife]$.

We have following simple and useful propositions:

Proposition 1 Any family relation must have at least one reverse relation.

This is like an axiom. In any language if there is a term for a family relation such as [Mother] then there must be a term for [Son] or [Daughter].

This means if party pair $(p_i, p_j) \in R_1$ then there must be a relation R_2 that $(p_j, p_i) \in R_2$ and so

$$R_2 = \mathrm{R}(R_1).$$

For primary family relations it is easy to find the reverse pair (p_j, p_i) for any given party pair (p_i, p_j) by searching in Table 4. It is easy to search the reverse pair by a simple query in a database as well. We will discuss about the database implementation for family relations in Section 7.

6. Derived Family Relations

Any relation can be denoted as a set of party pairs but sometimes, listing all pairs for all relations may be very time consuming. We will create an operation for relations to simplify this process and make the questions on family relations easier to be answered.

Suppose we have two relations based on the power set of a universe, $P(U)$:

$$R_1 = \{ (a_i, a_j) \mid a_i \in P(U),\ a_j \in P(U) \} \quad (23)$$

and

$$R_2 = \{ (b_k, b_l) \mid b_k \in P(U),\ b_l \in P(U) \} \quad (24)$$

Define two set functions: FROM function, denoted as F, for all FROM parties of R_1 and TO function, denoted as T, for all TO parties of R_2:

$$\mathrm{F}\ (R_1) = \{ a_i \mid a_i \in P(U) \} \quad (25)$$

$$\mathrm{T}\ (R_2) = \{ b_l \mid b_l \in P(U) \} \quad (26)$$

and assume

$$I = \mathrm{T}\ (R_2) \cap \mathrm{F}\ (R_1) \neq \emptyset \quad (27)$$

then the operation, called *connection* and denoted as ~, is defined as

$$R_2 \sim R_1 = \{ (c_m,\ c_n) \mid (c_m, d_s) \in R_2,\ (d_s, c_n) \in R_1,\ d_s \in I \} \quad (28)$$

The resulted relation by the connection operation is called a *derived* relation. In terms of party pairs, we can also denote:

$$(c_m, c_n) = (c_m,\ d_s) \sim (d_s,\ c_n) \quad (29)$$

where d_s is called a *connector*.

For example, in Table 3 (Party Table), p_{12} (Grandfather2) is p_3 (Son1)'s mother (p_2)'s father, therefore the party pair (p_{12}, p_2) is listed in relation [Father] in Table 4. Since

$$\mathrm{T}\ [Father] = \{ p_3, p_4, p_5, p_6, p_7, p_8, p_{10}, p_1, p_2 \} \quad (30)$$

$$\mathrm{F}\ [Mother] = \{ p_2, p_{14}, p_{15} \} \quad (31)$$

So,

$$I = \mathrm{T}\ [Father] \cap \mathrm{F}\ [Mother] = \{ p_2 \} \neq \emptyset \quad (32)$$

Now we can get a derived relation:

$$[Grandfather1] = [Father] \sim [Mother]$$
$$= \{ (p_{12}, p_2) \sim (p_2, p_3), (p_{12}, p_2) \sim (p_2, p_4), (p_{12}, p_2) \sim (p_2, p_5),$$
$$(p_{12}, p_2) \sim (p_2, p_6), (p_{12}, p_2) \sim (p_2, p_7), (p_{12}, p_2) \sim (p_2, p_8) \}$$
$$= \{ (p_{12}, p_3), (p_{12}, p_4), (p_{12}, p_5), (p_{12}, p_6), (p_{12}, p_7), (p_{12}, p_8) \} \quad (33)$$

By the common sense, father's father is also called grandfather and so we can get another derived relation:

$$[Grandfather2] = [Father] \sim [Father]$$
$$= \{ (p_{11}, p_1) \sim (p_1, p_3), (p_{11}, p_1) \sim (p_1, p_4), (p_{11}, p_1) \sim (p_1, p_5),$$
$$(p_{11}, p_1) \sim (p_1, p_6), (p_{11}, p_1) \sim (p_1, p_7), (p_{11}, p_1) \sim (p_1, p_8) \}$$
$$= \{ (p_{11}, p_3), (p_{11}, p_4), (p_{11}, p_5), (p_{11}, p_6), (p_{11}, p_7), (p_{11}, p_8) \} \quad (34)$$

Put (33) and (34) together by union both pair sets, we have

$$[Grandfather] = [Father] \sim [Mother] \cup [Father] \sim [Father]$$
$$= \{ (p_{12}, p_3), (p_{12}, p_4), (p_{12}, p_5), (p_{12}, p_6), (p_{12}, p_7), (p_{12}, p_8),$$
$$(p_{11}, p_3), (p_{11}, p_4), (p_{11}, p_5), (p_{11}, p_6), (p_{11}, p_7), (p_{11}, p_8) \} \quad (35)$$

Table 5 listed all derived relations based on the primary relations in Table 4.

Table 5. *Derived Relations Table.*

Relation	Formula
[Grandfather]	[Father]~[Father] ∪ [Father]~[Mother]
[Grandmother]	[Mother]~[Father] ∪ [Mother]~[Mother]
[Grandparent]	[Grandfather] ∪ [Grandmother] = [Father]~[Father] ∪ [Father]~[Mother] ∪ [Mother]~[Father] ∪ [Mother]~[Mother]
[Grandparents]	[Parents] ~ [Father] ∪ [Parents] ~ [Mother]
[Grandson]	[Son] ~ [Son] ∪ [Son] ~ [Daughter]
[Grandsons]	[Sons] ~ [Son] ∪ [Sons] ~ [Daughter] ∪ [Sons] ~ [Sons] ∪ [Sons] ~ [Daughters]
[Granddaughter]	[Daughter] ~ [Son] ∪ [Daughter] ~ [Daughter]
[Granddaughters]	[Daughters] ~ [Son] ∪ [Daughters] ~ [Daughter] ∪ [Daughters] ~ [Sons] ∪ [Daughters] ~ [Daughters]
[Grandchild]	[Son] ~ [Son] ∪ [Son] ~ [Daughter] ∪ [Daughter] ~ [Son] ∪ [Daughter] ~ [Daughter]
[Grandchildren]	[Children] ~ [Son] ∪ [Children] ~ [Daughter] ∪ [Children] ~ [Sons] ∪ [Children] ~ [Daughters]
[Father-in-law]	[Father] ~ [Husband] ∪ [Father] ~ [Wife]
[Mother-in-law]	[Mother] ~ [Husband] ∪ [Mother] ~ [Wife]
[Parent-in-law]	[Father] ~ [Husband] ∪ [Father] ~ [Wife] ∪ [Mother] ~ [Husband] ∪ [Mother] ~ [Wife]
[Parents-in-law]	[Parents] ~ [Husband] ∪ [Parents] ~ [Wife]
[Brother-in-law]	[Brother] ~ [Husband] ∪ [Brother] ~ [Wife]
[Brothers-in-law]	[Brothers] ~ [Husband] ∪ [Brothers] ~ [Wife]
[Sister-in-law]	[Sister] ~ [Husband] ∪ [Sister] ~ [Wife]
[Sisters-in-law]	[Sisters] ~ [Husband] ∪ [Sisters] ~ [Wife]
[Uncle]	[Brother] ~ [Father] ∪ [Brother] ~ [Mother] ∪ [Husband] ~ [Sister] ~ [Father] ∪ [Husband] ~ [Sister] ~ [Mother]
[Uncles]	[Brothers] ~ [Father] ∪ [Brothers] ~ [Moher] ∪ [Husbands] ~ [Sisters] ~ [Father] ∪ [Husbands] ~ [Sisters] ~ [Mother]
[Aunt]	[Sister] ~ [Father] ∪ [Sister] ~ [Moher] ∪ [Wife] ~ [Brother] ~ [Father] ∪ [Wife] ~ [Brother] ~ [Mother]
[Aunts]	[Sisters] ~ [Father] ∪ [Sisters] ~ [Moher] ∪ [Wives] ~ [Brothers] ~ [Father] ∪ [Wives] ~ [Brothers] ~ [Mother]
[Nephew]	[Son] ~ [Brother] ∪ [Son] ~ [Sister]
[Nephews]	[Sons] ~ [Brother] ∪ [Sons] ~ [Sister] ∪ [Sons] ~ [Brothers] ∪ [Sons] ~ [Sisters]
[Niece]	[Daughter] ~ [Brother] ∪ [Daughter] ~ [Sister]
[Nieces]	[Daughters] ~ [Brother] ∪ [Daughters] ~ [Sister] U [Daughters] ~ [Brothers] ∪ [Daughters] ~ [Sisters]
[Cousin]	[Son] ~ [Brother] ~ [Father] ∪ [Son] ~ [Sister] ~ [Father] ∪ [Son] ~ [Brother] ~ [Mother] ∪ [Son] ~ [Sister] ~ [Mother] ∪ [Daughter] ~ [Brother] ~ [Father] ∪ [Daughter] ~ [Sister] ~ [Father] ∪ [Daughter] ~ [Brother] ~ [Mother] ∪ [Daughter] ~ [Sister] ~ [Mother]
[Cousins]	[Sons] ~ [Brother] ~ [Father] ∪ [Sons] ~ [Sister] ~ [Father] ∪ [Sons] ~ [Brother] ~ [Mother] ∪ [Sons] ~ [Sister] ~ [Mother] ∪ [Daughters] ~ [Brother] ~ [Father] ∪ [Daughters] ~ [Sister] ~ [Father] ∪ [Daughters] ~ [Brother] ~ [Mother] ∪ [Daughters] ~ [Sister] ~ [Mother] ∪ [Children] ~ [Brother] ~ [Father] ∪ [Children] ~ [Sister] ~ [Father] ∪ [Children] ~ [Brother] ~ [Mother] ∪ [Children] ~ [Sister] ~ [Mother] ∪ [Sons] ~ [Brothers] ~ [Father] ∪ [Sons] ~ [Sisters] ~ [Father] ∪ [Sons] ~ [Brothers] ~ [Mother] ∪ [Sons] ~ [Sisters] ~ [Mother] ∪ [Daughters] ~ [Brothers] ~ [Father] ∪ [Daughters] ~ [Sisters] ~ [Father] ∪ [Daughters] ~ [Brothers] ~ [Mother] ∪ [Daughters] ~ [Sisters] ~ [Mother] ∪ [Children] ~ [Brothers] ~ [Father] ∪ [Children] ~ [Sisters] ~ [Father] ∪ [Children] ~ [Brothers] ~ [Mother] ∪ [Children] ~ [Sisters] ~ [Mother]

We have some propositions about the reverses of derived relations that is helpful to find out the reverse pair (p_j, p_i) for a given pair (p_i, p_j).

Proposition 2 For derived relations we have

$$\mathrm{R}\ (R_2 \sim R_1) = \mathrm{R}\ (R_1) \sim \mathrm{R}\ (R_2) \tag{36}$$

This is not hard to prove. In fact, from connection definition (28):

$R_2 \sim R_1 = \{\ (c_m\ ,\ \ c_n) \mid (c_m, d_s) \in R_2\ ,\ \ (\ d_s, c_n) \in R_1,\ d_s \in I\ \}$

Then

$$\mathrm{R}\ (R_2 \sim R_1) = \mathrm{R}\ \{\ (c_m,\ c_n) \mid (c_m, d_s) \in R_2,\ (d_s, c_n) \in R_1,\ d_s \in I\ \}$$

$$= \{\mathrm{R}\ (c_m, c_n) \mid (c_m, d_s) \in R_2, (d_s, c_n) \in R_1, d_s \in I\ \}$$

$$= \{\ (c_n, c_m) \mid (c_n, d_s) \in \mathrm{R}\ (R_1), (d_s, c_m) \in \mathrm{R}\ (R_2),\ d_s \in I\ \}$$

$$= \mathrm{R}\ (R_1) \sim \mathrm{R}\ (R_2)$$

This proposition can be also presented in terms of party pairs:

$$\mathrm{R}\ ((c_m, d_s) \sim (d_s, c_n)) = \mathrm{R}\ (d_s, c_n) \sim \mathrm{R}\ (c_m, d_s) = (c_n, d_s) \sim (d_s, c_m) \tag{37}$$

Proposition 3

$$\mathrm{R}\ (\sim \prod_{i=1}^{N} R_i) = \sim \prod_{i=N}^{1} \mathrm{R}\ (R_i) \tag{38}$$

where

$$\sim \prod_{i=1}^{N} R_i = R_1 \sim R_2 \sim \ldots \sim R_N \tag{39}$$

This is easy to be proved using a simple mathematical induction from Proposition 2. Suppose

$$\mathrm{R}\ \left(\sim \prod_{i=1}^{k} R_i\right) = \sim \prod_{j=k}^{1} \mathrm{R}\ (R_j), \text{ then for } k+1,$$

$$\mathrm{R}\ \left(\sim \prod_{i=1}^{k+1} R_i\right) = \mathrm{R}\ \left((\sim \prod_{i=1}^{k} R_i) \sim R_{k+1}\right) =$$

$$\mathrm{R}\ (R_{k+1}) \sim \mathrm{R}\ \left(\sim \prod_{i=1}^{k} R_i\right) = \mathrm{R}\ (R_{k+1}) \sim (\sim \prod_{i=k}^{1} \mathrm{R}\ (R_i)) = \sim \prod_{i=k+1}^{1} \mathrm{R}\ (R_j).$$

Note, reverse is not unique and so there might be multiple reverse relations for a given relation. However, with these propositions it is easier to find the derived relation that contains the reverse party pair (p_j, p_i) for a given pair (p_i, p_j) in a derived relation. We will use an example to show the steps below.

From Table 3 (Party Table), we have:

p_{12} = Grandfather2 (Wife's father)

p_6 = Daughter1

p_2 = Wife

and from Table 5 (Derived Relations), we have [Father] ~ [Mother] = [Grandfather]. Note, in Table 5, actually we have [Grandfather] = [Father] ~ [Father] ∪ [Father] ~ [Mother] but we only need the mother part [Father] ~ [Mother] because p_{12} = grandfather2 is mother's father.

The party pair $(p_{12}, p_6) \in$ [Grandfather]. Now the question is how to find the derived relation containing

$\mathrm{R}\ (p_{12}, p_6) = (p_6, p_{12})$

Since $(p_{12}, p_6) \in$ [Grandfather] = [Father] ~ [Mother] and so p_2 (= Wife) is the connector to connect (p_{12}, p_2) and (p_2, p_6), i.e. p_2 is p_6's mother and is p_{12}'s daughter as well. So, we have $(p_{12}, p_6) = (p_{12}, p_2) \sim (p_2, p_6)$. According to Proposition 2 formula (36) we have:

$$\mathrm{R}\ (p_{12}, p_6) = \mathrm{R}\ \left((p_{12}, p_2) \sim (p_2, p_6)\right)$$

$$= \mathrm{R}\ (p_2, p_6) \sim \mathrm{R}\ (p_{12}, p_2) = (p_6, p_2) \sim (p_2, p_{12})$$

Now, from Table 4 (Primary Family Relations) we know:

$$(p_6, p_2) \in [Daughter] \text{ and } (p_2, p_{12}) \in [Daughter]$$

Therefore

$$(p_6, p_{12}) = (p_6, p_2) \sim (p_2, p_{12}) \in [Daughter] \sim [Daughter] = [Granddaughter]$$

which means [Granddaughter] is the derived relation that contains $\mathrm{R}\ (p_{12}, p_6) = (p_6, p_{12})$.

Note, for human intelligence the reverse relation of [Grandfather] is obviously [Granddaughter] because p_6 = Daughter1. The point here is that for Logical Artificial Intelligence it needs to be implemented in computer tools with coding.

7. Database Implementation

The theoretical descriptions discussed above are not good for calculations in Logical Artificial Intelligence (LAI) using any computer languages. The best way for easier calculations is to implement them in a relational database such as MS SQL Server, Oracle and etc. which are very popular in the IT world. The following are relational database (DB) implementations in MS SQL Server.

The two major parts in DB implementations are tables and queries. Tables are for storing data and queries are for retrieving data.

The two major tables for Family Relations are listed below. Table 6 is for Primary Relations and Table 7 is for Derived Relations.

7.1. Primary Relations Table

Table 6. DB Table tblPrimaryRelations.

rID	Relation	FromParty	ToParty	Description
1	Husband	p1	p2	p1 is p2’s husband
2	Husband	p11	p14	p11 is p14’s husband
3	Husband	p12	p15	p12 is p15’s husband
4	Wife	p2	p1	p2 is p1’s wife
5	Wife	p14	p11	p14 is p11’s wife
6	Wife	p15	p12	p15 is p12’s wife
7	Father	p1	p3	p1 is p3's father
8	Father	p1	p4	p1 is p4's father
9	Father	p1	p5	p1 is p5's father
10	Father	p1	p6	p1 is p6's father
11	Father	p1	p7	p1 is p7's father
12	Father	p1	p8	p1 is p8's father
13	Father	p1	p10	p1 is p10's father
14	Father	p11	p1	p11 is p1's father
15	Father	p12	p2	p12 is p2's father
16	Mother	p2	p3	p2 is p3's mother
17	Mother	p2	p4	p2 is p4's mother
18	Mother	p2	p5	p2 is p5's mother
19	Mother	p2	p6	p2 is p6's mother
20	Mother	p2	p7	p2 is p7's mother
21	Mother	p2	p8	p2 is p8's mother
22	Mother	p2	p10	p2 is p10's mother
23	Mother	p14	p1	p14 is p1's mother
24	Mother	p15	p2	p15 is p2's mother
25	Son	p3	p1	p3 is p1's son
26	Son	p3	p2	p3 is p2's son
27	Son	p4	p1	p4 is p1's son
28	Son	p4	p2	p4 is p2's son
29	Son	p1	p11	p1 is p11's son
30	Son	p1	p14	p1 is p14's son
31	Son	p1	p18	p1 is p18's son
32	Sons	p5	p1	p5 are p1's sons
33	Sons	p5	p2	p5 are p2's sons
34	Daughter	p6	p1	p6 is p1's daughter
35	Daughter	p6	p2	p6 is p2's daughter

rID	Relation	FromParty	ToParty	Description
36	Daughter	p7	p1	p7 is p1's daughter
37	Daughter	p7	p2	p7 is p2's daughter
38	Daughter	p2	p12	p2 is p12's daughter
39	Daughter	p2	p15	p2 is p15's daughter
40	Daughter	p2	p19	p2 is p19's daughter
41	Daughtes	p8	p1	p8 are p1's daughters
42	Daughtes	p8	p2	p8 are p2's daughters
43	Parent	p1	p3	p1 is p3's parent
44	Parent	p1	p4	p1 is p4's parent
45	Parent	p1	p5	p1 is p5's parent
46	Parent	p1	p6	p1 is p6's parent
47	Parent	p1	p7	p1 is p7's parent
48	Parent	p1	p8	p1 is p8's parent
49	Parent	p1	p10	p1 is p10's parent
50	Parent	p11	p1	p11 is p1's parent
51	Parent	p12	p2	p12 is p2's parent
52	Parent	p2	p3	p2 is p3's parent
53	Parent	p2	p4	p2 is p4's parent
54	Parent	p2	p5	p2 is p5's parent
55	Parent	p2	p6	p2 is p6's parent
56	Parent	p2	p7	p2 is p7's parent
57	Parent	p2	p8	p2 is p8's parent
58	Parent	p2	p10	p2 is p10's parent
59	Parent	p14	p1	p14 is p1's parent
60	Parent	p15	p2	p15 is p2's parent
61	Parents	p9	p3	p9 are p3's parents
62	Parents	p9	p4	p9 are p4's parents
63	Parents	p9	p5	p9 are p5's parents
64	Parents	p9	p6	p9 are p6's parents
65	Parents	p9	p7	p9 are p7's parents
66	Parents	p9	p8	p9 are p8's parents
67	Parents	p9	p10	p9 are p10's parents
68	Parents	p18	p1	p18 are p1's parents
69	Parents	p19	p2	p19 are p2's parents
70	Child	p3	p1	p3 is p1's child
71	Child	p3	p2	p3 is p2's child
72	Child	p4	p1	p4 is p1's child
73	Child	p4	p2	p4 is p2's child
74	Child	p6	p1	p6 is p1's child

rID	Relation	FromParty	ToParty	Description
75	Child	p6	p2	p6 is p2's child
76	Child	p7	p1	p7 is p1's child
77	Child	p7	p2	p7 is p2's child
78	Child	p1	p11	p1 is p11's child
79	Child	p1	p14	p1 is p14's child
80	Child	p1	p18	p1 is p18's child
81	Child	p2	p12	p2 is p12's child
82	Child	p2	p15	p2 is p15's child
83	Child	p2	p19	p2 is p19's child
84	Children	p10	p1	p10 are p1's children
85	Children	p10	p2	p10 are p2's children
86	Children	p10	p9	p10 are p9's children
87	Children	p5	p1	p5 are p1's children
88	Children	p5	p2	p5 are p2's children
89	Children	p5	p9	p5 are p9's children
90	Children	p8	p1	p8 are p1's children
91	Children	p8	p2	p8 are p2's children
92	Children	p8	p9	p8 are p9's children
93	Brother	p3	p4	p3 is p4's brother
94	Brother	p3	p6	p3 is p6's brother
95	Brother	p3	p7	p3 is p7's brother
96	Brother	p3	p8	p3 is p8's brother
97	Brother	p4	p3	p4 is p3's brother
98	Brother	p4	p6	p4 is p6's brother
99	Brother	p4	p7	p4 is p7's brother
100	Brother	p4	p8	p4 is p8's brother
101	Brothers	p5	p6	p5 are p6's brothers
102	Brothers	p5	p7	p5 are p7's brothers
103	Brothers	p5	p8	p5 are p8's brothers
104	Sister	p6	p3	p6 is p3's sister
105	Sister	p6	p4	p6 is p4's sister
106	Sister	p6	p5	p6 is p5's sister
107	Sister	p6	p7	p6 is p7's sister
108	Sister	p7	p3	p7 is p3's sister
109	Sister	p7	p4	p7 is p4's sister
110	Sister	p7	p5	p7 is p5's sister
111	Sister	p7	p6	p7 is p6's sister
112	Sisters	p8	p3	p8 are p3's sisters
113	Sisters	p8	p4	p8 are p4's sisters

rID	Relation	FromParty	ToParty	Description
114	Sisters	p8	p5	p8 are p5's sisters
115	Grandfathers	p13	p3	p13 are p3's grandfathers
116	Grandfathers	p13	p4	p13 are p4's grandfathers
117	Grandfathers	p13	p5	p13 are p5's grandfathers
118	Grandfathers	p13	p6	p13 are p6's grandfathers
119	Grandfathers	p13	p7	p13 are p7's grandfathers
120	Grandfathers	p13	p8	p13 are p8's grandfathers
121	Grandfathers	p13	p10	p13 are p10's grandfathers
122	Grandmothers	p16	p3	p16 are p3's grandmothers
123	Grandmothers	p16	p4	p16 are p4's grandmothers
124	Grandmothers	p16	p5	p16 are p5's grandmothers
125	Grandmothers	p16	p6	p16 are p6's grandmothers
126	Grandmothers	p16	p7	p16 are p7's grandmothers
127	Grandmothers	p16	p8	p16 are p8's grandmothers
128	Grandmothers	p16	p10	p16 are p10's grandmothers
129	Grandparents	p17	p3	p17 are p3's grandparents
130	Grandparents	p17	p4	p17 are p4's grandparents
131	Grandparents	p17	p5	p17 are p5's grandparents
132	Grandparents	p17	p6	p17 are p6's grandparents
133	Grandparents	p17	p7	p17 are p7's grandparents
134	Grandparents	p17	p8	p17 are p8's grandparents
135	Grandparents	p17	p10	p17 are p10's grandparents

In tblPrimaryRelations, the column [FromParty] is for the first party in the relation pair and the column [ToParty] is for the second party in the relation pair. For example, the relation pair (p_2, p_3) means p_2 is p_3's mother. This relation is from p_2 to p_3 and so we call p_2 as a *From Party* and p_3 as a *To Party*. By the way, in DB we cannot type subscriptions and so we use p2 for p_2, p3 for p_3, and so forth.

Another thing we need to mention is to distinguish synonyms (such as Parent) and group relations (such as Parents). Parent is a synonym relation to Father or Mother and so it is a union of Father relation set and Mother relation set. Parents means both Father and Mother which is a group of two people and so Parents is a group party defined in the Party Table. It composes relation pairs with other parties (Person or Group) independently.

7.2. Derived Relations Table

Table 7. *DB Table tblDerivedRelations.*

dID	Relation	LeftRelation	RightRelation	Description
1	Grandfather	Father	Father	Father's father
2	Grandfather	Father	Mother	Mother's father
3	Grandfather	Father	Uncle	Uncle's father = Father's Father = Mother's Father

dID	Relation	LeftRelation	RightRelation	Description
4	Grandfather	Father	Aunt	Aunt's father = Father's Father = Mother's Father
5	Grandmother	Mother	Father	Father's mother
6	Grandmother	Mother	Mother	Mother's mother
7	Grandmother	Mother	Uncle	Uncle's mother = Father's mother = Mother's mother
8	Grandmother	Mother	Aunt	Aunt's mother = Father's mother = Mother's mother
9	Grandparent	Parent	Father	Father's parent
10	Grandparent	Parent	Mother	Mother's parent
11	Grandparent	Parent	Uncle	Uncle's parent = Father's parent = Mother's parent
12	Grandparent	Parent	Aunt	Aunt's parent = Father's parent = Mother's parent
13	Grandparents	Parents	Father	Father's parents
14	Grandparents	Parents	Mother	Mother's parents
15	Grandparents	Parents	Uncle	Uncle's parents = Father's parents = Mother's parents
16	Grandparents	Parents	Aunt	Aunt's parents = Father's parents = Mother's parents
17	Grandson	Son	Son	Son's son
18	Grandson	Son	Daughter	Daughter's son
19	Grandsons	Sons	Son	Son's sons
20	Grandsons	Sons	Daughter	Daughter's sons
21	Grandsons	Sons	Sons	Sons' sons
22	Grandsons	Sons	Daughters	Daughters' sons
23	Granddaughter	Daughter	Son	Son's daughter
24	Granddaughter	Daughter	Daughter	Daughter's daughter
25	Granddaughters	Daughters	Son	Son's daughters
26	Granddaughters	Daughters	Daughter	Daughter's daughters
27	Granddaughters	Daughters	Sons	Sons' daughters
28	Granddaughters	Daughters	Daughters	Daughters' daughters
29	Grandchild	Son	Son	Son's child
30	Grandchild	Son	Daughter	Daughter's child
31	Grandchild	Daughter	Son	Son's child
32	Grandchild	Daughter	Daughter	Daughter's child
33	Grandchildren	Children	Son	Son's children
34	Grandchildren	Children	Daughter	Daughter's children
35	Grandchildren	Children	Sons	Sons' children
36	Grandchildren	Children	Daughters	Daughters' children
37	Father-in-law	Father	Husband	Husband's father
38	Father-in-law	Father	Wife	Wife's father
39	Mother-in-law	Mother	Husband	Husband's mother
40	Mother-in-law	Mother	Wife	Wife's mother
41	Parent-in-law	Father	Husband	Husband's father
42	Parent-in-law	Father	Wife	Wife's father

dID	Relation	LeftRelation	RightRelation	Description
43	Parent-in-law	Mother	Husband	Husband's mother
44	Parent-in-law	Mother	Wife	Wife's mother
45	Parents-in-law	Parents	Husband	Husband's parents
46	Parents-in-law	Parents	Wife	Wife's parents
47	Brother-in-law	Brother	Husband	Husband's brother
48	Brother-in-law	Brother	Wife	Wife's brother
49	Brothers-in-law	Brothers	Husband	Husband's brothers
50	Brothers-in-law	Brothers	Wife	Wife's brothers
51	Sister-in-law	Sister	Husband	Husband's sister
52	Sister-in-law	Sister	Wife	Wife's sister
53	Sisters-in-law	Sisters	Husband	Husband's sisters
54	Sisters-in-law	Sisters	Wife	Wife's sisters
55	Uncle	Brother	Father	Father's brother
56	Uncle	Brother	Mother	Mother's brother
57	Uncle	Husband	Aunt	Aunt's husband
58	Uncles	Brothers	Father	Father's brothers
59	Uncles	Brothers	Mother	Mother's brothers
60	Uncles	Husbands	Aunts	Aunts' husbands
61	Aunt	Sister	Father	Father's sister
62	Aunt	Sister	Mother	Mother's sister
63	Aunt	Wife	Uncle	Uncle's wife
64	Aunts	Sisters	Father	Father's sisters
65	Aunts	Sisters	Mother	Mother's sisters
66	Aunts	Wives	Uncles	Uncles' wives
67	Nephew	Son	Brother	Brother's son
68	Nephew	Son	Sister	Sister's son
69	Nephews	Sons	Brother	Brother's sons
70	Nephews	Sons	Sister	Sister's sons
71	Nephews	Sons	Brothers	Brothers' sons
72	Nephews	Sons	Sisters	Sisters' sons
73	Niece	Daughter	Brother	Brother's daughter
74	Niece	Daughter	Sister	Sister's daughter
75	Nieces	Daughters	Brother	Brother's daughters
76	Nieces	Daughters	Sister	Sister's daughters
77	Nieces	Daughters	Brothers	Brothers' daughters
78	Nieces	Daughters	Sisters	Sisters' daughters
79	Cousin	Son	Uncle	Uncle's son
80	Cousin	Son	Aunt	Aunt's son
81	Cousin	Daughter	Uncle	Uncle's daughter

dID	Relation	LeftRelation	RightRelation	Description
82	Cousin	Daughter	Aunt	Aunt's daughter
83	Cousins	Sons	Uncle	Uncle's sons
84	Cousins	Sons	Aunt	Aunt's sons
85	Cousins	Daughters	Uncle	Uncle's daughters
86	Cousins	Daughters	Aunt	Aunt's daughters
87	Cousins	Children	Uncle	Uncle's children
88	Cousins	Children	Aunt	Aunt's children
89	Cousins	Sons	Uncles	Uncles' sons
90	Cousins	Sons	Aunts	Aunts' sons
91	Cousins	Daughters	Uncles	Uncle's daughters
92	Cousins	Daughters	Aunts	Aunts' daughters
93	Cousins	Children	Uncles	Uncles' children
94	Cousins	Children	Aunts	Aunts' children
95	Son-in-law	Husband	Daughter	Daughter's Husband
96	Daughter-in-law	Wife	Son	Son's Wife

In tblDerivedRelations, the column [LeftRelation] means the relation on the left side of the connection symbol ~ and [RightRelation] means the relation on the right side of the connection symbol ~. For example, [Grandfather] = [Father] ~ [Mother], meaning mother's father is called Grandfather.

tblPrimaryRelations and tblDerivedRelations are the two major tables in the DB. But for answering the user questions we need more tables, listed below.

7.3. Parties Table

Table 8. DB Table tblParties.

pID	Party	Role	Description	Gender	FirstName	LastName
1	p1	Husband	Person	M		
2	p2	Wife	Person	F		
3	p3	Son1	Person	M		
4	p4	Son2	Person	M		
5	p5	Sons	Group {p3, p4}	M		
6	p6	Daughter1	Person	F		
7	p7	Daughter2	Person	F		
8	p8	Daughters	Group {p6, p7}	F		
9	p9	Parents	Group {p1, p2}	MF		
10	p10	Children	Group {p3, p4, p6, p7}	MF		
11	p11	Grandfather1	Person - Husband's father	M		
12	p12	Grandfather2	Person - Wife's father	M		

pID	Party	Role	Description	Gender	FirstName	LastName
13	p13	Grandfathers	Group {p11, p12}	M		
14	p14	Grandmother1	Person - Husband's mother	F		
15	p15	Grandmother2	Person - Wife's mother	F		
16	p16	Grandmothers	Group {p14, p15}	F		
17	p17	Grandparents	Group {p11, p12, p14, p15}	MF		
18	p18	Grandparents1	Group - Husband's parents {p11, p14}	MF		
19	p19	Grandparents2	Group - Wife's parents {p12, p15}	MF		

tblParties is a very basic table for composing tblPrimaryRelations because primary relations are all based on parties defined in this tblParties. In this table [Party] is the most important column that identifies each specific party, but all those parties like p1, p2 do not tell us who they are, therefore we use a supplementary column [Role] to explain the role this party is playing in the family. Note, Role is relevant and party is absolute. For example, p1 is a man and is the husband of his wife, then we use "Husband" as his role. You may change his role to "Father" because he is also his children's father. To avoid confusions, we simply use one term to indicate the role of the party. The accurate relations are all defined in tblPrimaryRelations in party pairs.

[Gender] is a supplemental column to help identifying the gender of the party. Also note, we use numbers to distinguish the parties with the same role. For example, Son1 and Son2 are two sons and usually we assume the age of Son1 is greater than Son2. But we do not use more than 2 roles for the same kind of role. So, we do not define Son3 and Son4 and etc. In case a user question includes more than 2 roles, say 5 sons, we can handle it by adding more temporary roles in tblParties by coding. In that case we will need to add corresponding party pairs in tblPrimaryRelations. We have an example in Section 8 (Example Questions and SQL Queries, Question 4) to show how to add temporary roles in tblParties.

[FirstName] and [LastName] are two supplemental columns in case we need to specify names for one or more parties. We may add more temporary columns in this table to answer some specific questions. All temporary columns and rows will be deleted automatically by the database administration.

7.4. Synonyms Table

Table 9. DB Table tblSynonyms.

sID	Synonym	KeyRelation	RelationTable	NumOfKeys
1	Brother	Brother	tblPrimaryRelations	1
2	Brothers	Brothers	tblPrimaryRelations	1
3	Child	Son	tblPrimaryRelations	2
4	Child	Daughter	tblPrimaryRelations	2
5	Children	Children	tblPrimaryRelations	1
6	Daughter	Daughter	tblPrimaryRelations	1
7	Daughters	Daughters	tblPrimaryRelations	1
8	Father	Father	tblPrimaryRelations	1
9	Dad	Father	tblPrimaryRelations	1
10	Daddy	Father	tblPrimaryRelations	1
11	Husband	Husband	tblPrimaryRelations	1
12	Mother	Mother	tblPrimaryRelations	1
13	Mom	Mother	tblPrimaryRelations	1

sID	Synonym	KeyRelation	RelationTable	NumOfKeys
14	Mommy	Mother	tblPrimaryRelations	1
15	Parent	Father	tblPrimaryRelations	2
16	Parent	Mother	tblPrimaryRelations	2
17	Parents	Parents	tblPrimaryRelations	1
18	Sister	Sister	tblPrimaryRelations	1
19	Sisters	Sisters	tblPrimaryRelations	1
20	Son	Son	tblPrimaryRelations	1
21	Sons	Sons	tblPrimaryRelations	1
22	Wife	Wife	tblPrimaryRelations	1
23	Aunt	Aunt	tblDerivedRelations	1
24	Aunts	Aunts	tblDerivedRelations	1
25	Cousin	Cousin	tblDerivedRelations	1
26	Cousins	Cousins	tblDerivedRelations	1
27	Grandchild	Grandson	tblDerivedRelations	2
28	Grandchild	Granddaughter	tblDerivedRelations	2
29	Grandchildren	Grandchildren	tblDerivedRelations	1
30	Granddaughter	Granddaughter	tblDerivedRelations	1
31	Granddaughters	Granddaughters	tblDerivedRelations	1
32	Grandfather	Grandfather	tblDerivedRelations	1
33	Grandpa	Grandfather	tblDerivedRelations	1
34	Grandpas	Grandfathers	tblDerivedRelations	1
35	Grandfathers	Grandfathers	tblDerivedRelations	1
36	Grandmother	Grandmother	tblDerivedRelations	1
37	Grandma	Grandmother	tblDerivedRelations	1
38	Grandmas	Grandmothers	tblDerivedRelations	1
39	Grandmothers	Grandmothers	tblDerivedRelations	1
40	Grandparent	Grandfather	tblDerivedRelations	2
41	Grandparent	Grandmother	tblDerivedRelations	2
42	Grandparents	Grandparents	tblDerivedRelations	1
43	Grandson	Grandson	tblDerivedRelations	1
44	Grandsons	Grandsons	tblDerivedRelations	1
45	Nephew	Nephew	tblDerivedRelations	1
46	Nephews	Nephews	tblDerivedRelations	1
47	Niece	Niece	tblDerivedRelations	1
48	Nieces	Nieces	tblDerivedRelations	1
49	Uncle	Uncle	tblDerivedRelations	1
50	Uncles	Uncles	tblDerivedRelations	1
51	Brother-in-law	Brother-in-law	tblDerivedRelations	1
52	Brothers-in-law	Brothers-in-law	tblDerivedRelations	1

sID	Synonym	KeyRelation	RelationTable	NumOfKeys
53	Father-in-law	Father-in-law	tblDerivedRelations	1
54	Mother-in-law	Mother-in-law	tblDerivedRelations	1
55	Parent-in-law	Father-in-law	tblDerivedRelations	2
56	Parent-in-law	Mother-in-law	tblDerivedRelations	2
57	Parents-in-law	Parents-in-law	tblDerivedRelations	1
58	Sister-in-law	Sister-in-law	tblDerivedRelations	1
59	Sisters-in-law	Sisters-in-law	tblDerivedRelations	1
60	Son-in-law	Son-in-law	tblDerivedRelations	1
61	Daughter-in-law	Daughter-in-law	tblDerivedRelations	1

This table tblSynonyms is to help dealing with synonyms issue. A relation may have multiple names in the same language. For example, people call Mother as Mom or Mommy, call Father as Dad or Daddy. In more complicated case, someone may say "This is my parent" to introduce her mother or father to her friends. But for both mother and father, she should say "These are my parents". Here we need to distinguish the two relations Parent and Parents. Parent is only a synonym to Father or Mother, but not both, while Parents mean both Father and Mother, therefore Parents are a group of people. As an independent party Parents is a Group Party listed in tblParties. But Parent is not an independent party and cannot be listed in tblParties.

Each synonym has a corresponding Key relation listed in tblPrimaryRelations. For example, [Mom] has a key relation [Mother] which can be found in tblPrimaryRelations. Some synonyms may have more than one key relations. For example, [Parent] has two key relations [Father] or [Mother]. For this reason, we list [Parent] in tblPrimaryRelations as the union of [Father] and [Mother].

7.5. First Names Table

Table 10. DB Table tblPopularFirstNames.

FirstName	Gender
Aaliyah	F
Abigail	F
Adam	M
Addison	F
Aiden	M
Alex	M
Alexander	M
Alexis	F
Alice	F
Amber	F
Amelia	F
Angus	M
Anna	F
Annabelle	F
Archer	M
Archie	M
Aria	F
Ariana	F
Arlo	M
Arthur	M
Asher	M
Ashton	M
Athena	F
Aurora	F
Austin	M
Ava	F
Ayla	F
Beau	M
Beauden	M
Bella	F
Benjamin	M
Billie	F
Blake	M

FirstName	Gender
Bodhi	M
Braxton	M
Brooklyn	F
Caleb	M
Carter	M
Charles	M
Charlie	M
Charlie	F
Charlotte	F
Chloe	F
Connor	M
Cooper	M
Cora	F
Daisy	F
Daniel	M
David	M
Dylan	M
Eden	F
Edward	M
Eleanor	F
Eli	M
Elijah	M
Eliza	F
Elizabeth	F
Ella	F
Ellie	F
Elsie	F
Emilia	F
Emily	F
Emma	F
Ethan	M
Eva	F
Evelyn	F
Evie	F
Ezra	M
Felix	M
Finn	M
Fletcher	M
Florence	F

FirstName	Gender
Flynn	M
Frankie	F
Freya	F
Gabriel	M
George	M
Georgia	F
Grace	F
Grayson	M
Hannah	F
Harley	M
Harper	F
Harriet	F
Harrison	M
Harry	M
Harvey	M
Hazel	F
Heidi	F
Henry	M
Hudson	M
Hugo	M
Hunter	M
Imogen	F
Indie	F
Isaac	M
Isabella	F
Isabelle	F
Isla	F
Ivy	F
Jack	M
Jackson	M
Jacob	M
James	M
Jasmine	F
Jasper	M
Jaxon	M
Jayden	M
Jessica	F
John	M
Jordan	M

FirstName	Gender	FirstName	Gender
Joseph	M	Nixon	M
Joshua	M	Noah	M
Kaia	F	Olive	F
Kayden	M	Oliver	M
Kiara	F	Olivia	F
Kingston	M	Oscar	M
Lachlan	M	Paige	F
Layla	F	Patrick	M
Leah	F	Penelope	F
Leo	M	Peyton	F
Levi	M	Phoebe	F
Liam	M	Phoenix	M
Lilly	F	Piper	F
Lily	F	Poppy	F
Lincoln	M	Quinn	M
Logan	M	Quinn	F
Louie	M	Riley	M
Louis	M	Riley	F
Luca	M	Roman	M
Lucas	M	Rose	F
Lucy	F	Ruby	F
Luka	M	Ryan	M
Luke	M	Ryder	M
Luna	F	Sadie	F
Mackenzie	F	Samuel	M
Maddison	F	Sarah	F
Madison	F	Scarlett	F
Maia	F	Sebastian	M
Mason	M	Sienna	F
Matilda	F	Sofia	F
Matthew	M	Sophia	F
Max	M	Sophie	F
Maya	F	Stella	F
Mia	F	Summer	F
Michael	M	Thea	F
Mila	F	Theo	M
Millie	F	Theodore	M
Nathan	M	Thomas	M
Nina	F	Toby	M

FirstName	Gender
Tyler	M
Victoria	F
Violet	F
William	M
Willow	F
Wyatt	M
Xavier	M
Zachary	M

FirstName	Gender
Zara	F
Zoe	F
Zoey	F

This table comes from [12] for identifying the gender with a popular first name because different genders may have different relation names. For instance, Father means a male and Mother means a female. So sometimes we need gender information to determine a relation name.

7.6. Output Table

Table 11. *DB Table tblOutput.*

ID	Query	StepNum	OutputVariable	OutputValue
3	Query1.1	1	@Gender	M
1	Query1.2	2	@Party	P10
2	Query1.3	3	@ToParty	p1
4	Query1.4	4	[FirstName]	Sebastian
7	Query1.5	5	@ToParty	p10
8	Query1.5	5	@FromParty	p1
9	Query1.6	6	@Relation	Father
10	Query1.6	6	@Conclusion	p1 is p10's father
6	Query1.7	7	@FirstName	Sebastian
13	Query2.1	1	@Relation	Uncle
14	Query2.2	2	@Relation	Cousin
15	Query3.1	1	@Relation	Uncle
16	Query3.2	2	@Relation	Grandfather

This is a working table to hold the intermediate output results and provide input values for further processors and queries. We will show how to use this table in the Infobody Charts for the queries to answer the example questions in Section 8 (Example Questions and SQL Queries with Infobody Charts).

8. Example Questions and SQL Queries with Infobody Charts

Question 1 (Q1): What is the first name of the father of Sebastian's children?

Question 2 (Q2): What do you call your mother's brother's daughter?

Question 3 (Q3): What do you call your mother's brother's father?

Question 4 (Q4): (Missing Role Example) You have 3 sisters and you are the only one brother of them. What should your third sister's son call you?

The queries for Q1-Q4 will be listed below, but note, the SQL queries for Logical AI are based on Infobody Modeling (see [7] Chapter 3: Infobody Model in Terms of Graph Theory). The basic idea of infobody modeling is to separate processors (such as queries) as much as possible and specify input infobodies and output infobodies in details. This approach is different from regular SQL queries in terms of query structure. In a regular SQL query, the developer is often trying

to put all code in a single query or procedure to let the code complete the tasks as many as possible. But for Logical AI, we need to try to break down the code as much as possible to make each small piece of code doing only one single small task and get a single output. In terms of infobodies, we code each processor (query) as small as possible and let it generates only one (or two) output infobody based on necessary input infobodies. The infobody structure for each processor with the related input infobodies and output infobodies is called a *unit structure for the processor* in the infobody model. For Logical AI, we want to make a minimum number of output infobodies in an unit structure, in most cases, only one or two output infobodies. The answer to a user question usually needs a much bigger infobody structure composed with multiple unit structures from the original input infobodies to the final output infobodies indicating the answer. This infobody structure is called an *answer structure*. In this way, a unit structure may join multiple answer structures to answer different questions.

We will show two examples (Q2 and Q3) that use a common unit structure for two answer structures.

8.1. Question 1

Question 1 (Q1): What is the first name of the father of Sebastian's children?

SQL Code for Q1

```
-- For Question 1 (Q1): What is the first name of the father of Sebastian's children?
-- unit structures:

-- Query1.1: Get gender for 'Sebastian'
declare @Gender char(1)
select @Gender=[Gender] from [dbo].[tblPopularFirstNames]
where [FirstName] = 'Sebastian'

insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query1', 1, '@Gender', @Gender)
-- M

----------
-- Query1.2: 'Children' => [Party table].[role].[children] => [Party].[p10]
declare @Party varchar(10)
select @Party=Party from tblParties where Role = 'Children'

insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query1', 2, '@Party', @Party)
-- P10

--------------------
--Query1.3: 'Children' => [PrimaryRelation table].[Relation] => (p10, p1) U (p10, p2)
declare @Gender varchar(1)
select @Gender = [OutputValue] from [dbo].[tblOutput] where [StepNum] = 1

declare @ToParty varchar(10)
select @ToParty = ToParty from tblPrimaryRelations r
inner join [dbo].[tblParties] p on r.[ToParty] = p.[Party]
where r.Relation = 'Children' and substring(p.[Description], 1, 1) = 'P'
and p.Gender = @Gender -- M

print @ToParty -- p1
Print @Gender -- M

insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query2', 3, '@ToParty', @ToParty)
-- p1

--------------------
--Query1.4: Set [Party table].[p1].[FirstName] = 'Sebastian'
update [dbo].[tblParties] set [FirstName] = 'Sebastian' where [Party] = 'P1'
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query4', 4, '[FirstName]', 'Sebastian')
-- (1 row(s) affected) OK

----------
--Query1.5: (p10, p1) Get Reverse = (p1, p10) or R(p10, p1) = (p1, p10)
-- FromParty = 'p10' and ToParty = 'p1' Reverse is: FromParty = 'p1' and ToParty = 'p10'
-- This is not a real query but still need to put into the chart as Step 5

-- Query1.5 as Real SQL statement
declare @ToParty varchar(10)
declare @FromParty varchar(10)
select @ToParty = FromParty, @FromParty = ToParty from tblPrimaryRelations
where FromParty = 'p10' and ToParty = 'p1' and Relation = 'Children'

print @ToParty
-- p10
print @FromParty
-- p1

-- put into output table as 2 rows
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query5', 5, '@ToParty', @ToParty)
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query5', 5, '@FromParty', @FromParty)
-- OK
```

```
---------------
-- Query1.6: Get relation that contains (p1, p10) - it is in [primaryRelations].[Father]
-- join tblSynonyms
declare @Relation as varchar(50)
select @Relation = r.[Relation]
from [dbo].[tblPrimaryRelations] r
inner join [dbo].[tblParties] p on r.FromParty = p.Party
inner join [dbo].[tblSynonyms] s on r.Relation = s.[Synonym]
where r.[FromParty] = 'p1' and r.[ToParty] = 'p10' and sub-string(p.[Description], 1, 1) = 'P'
and s.NumOfKeys = 1
-- Father

declare @Conclusion as varchar(50)
set @Conclusion =
case @Relation
when 'Father' then 'p1 is p10''s father'
else 'p1 is not p10''s father'
end

print @Relation -- Father
print @Conclusion -- p1 is p10's father


-- put in tblOutputy
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query6', 6, '@Relation', @Relation)
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query6', 6, '@Conclusion', @Conclusion)
```

```
-------------------
--Query1.7: Get [Party table].[p1].[FirstName] = ‘Sabastian’
declare @FirstName varchar(50)
select @FirstName = [FirstName] from [dbo].[tblParties] where [Party] = 'P1'
--Sebastian
-- This is the answer to Question1
-- put in tblOutputy
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query7', 7, '@FirstName', @FirstName)

-- check output
select * from [dbo].[tblOutput] order by [StepNum]
```

Infobody Chart for Q1

Infobody chart is a visualization of infobody model. Input infobodies and output infobodies are presented in rectangulars and processors are presented in rounded rectangulars. The infobodies and processors are usually in containing chains indicating the detailed locations. The infobodies and processors are connected by arrows indicating the directions of input and output. The colors of boxes for infobodies and processors are for distinguishing each from other and no any other specific meaning and usually the infobodies at the same level have the same color. For more information about infobody charting, see [7] (Chapter2: Infobody Charting).

We will show the infobody charts by questions. Figures 2 and 3 below are for Q1.

Comparing SQL code Query1.1 and Figure 2, we can describe the unit structure for Query1.1 in infobody expression ([7] Section 2.1: Infobody Expressions) as follows:

$$[tblPopularFirstNames].([Gender],[FirstName].[Sebastian])$$

$$\rightarrow \langle Query1.1\rangle \Rightarrow$$

$$[tblOutput].([OutputVariable].[@Gender],[OutputValue].[M]) \quad (40)$$

and the final unit structure for Query1.7:

$$[tblParties].([Party].[p1],[FirstName].[Sebastian])$$

$$\rightarrow \langle Query1.7\rangle \Rightarrow$$

$$[tblOutput].([\ OutputVariable].[@FirstName],[OutputValue].[Sebastian]) \quad (41)$$

To simplify the infobody expressions and focus on the unit structures and connections to compose the answer structure, we can extend the infobody expressions by introducing new symbols as follows.

We will use:}UStr{: to present a unit structure named UStr including the processor and the related input and output infobodies, and then use $\xrightarrow{[Output]}$ to connect to the next unit structures indicating the major one or two output infobodies from the previous unit structure as the input infobodies to the next unit structure. If we have more than one unit structures (say 3) which have output infobodies for the same next unit

structure as the input infobodies, we will put the 3 unit structures into parentheses with comma (,) to separate them and each with one or two output infobodies on the arrow.

After the close parenthesis ")", we will use a double arrow ⇒ to connect to the next unit structure. This way we can simply connect all unit structures for Q1 to compose the answer structure for Q1 as shown in formula (42). Note, we only use the query numbers to identify the unit structures, for example, 1.1 means Query1.1 and so unit structure:}1.1{: means unit structure:}Query1.1{:.

$$(:\}1.1\{: \xrightarrow{M}:\}1.3\{: \xrightarrow{p1}:\}1.4\{: \xrightarrow{Sebastian},(:\}1.1\{: \xrightarrow{M}:\}1.3\{: \xrightarrow{p1},:\}1.2\{: \xrightarrow{p10})\Rightarrow:\}1.5\{: \xrightarrow{p1,p10}:\}1.6\{: \xrightarrow{p1\ is\ p10's\ father})$$

$$\Rightarrow:\}1.7\{: \xrightarrow{Sebastian}[\text{Answer} = \text{Sebastian}] \quad (42)$$

Formula (42) simplifies Figures 2 and 3 to a much simpler infobody expression for the answer structure of Q1. Note, we may separate each parenthesis for a single line to make it easier to read, but it is not necessary for infobody expressions. We can also put them together as a single line or two lines if necessary, as shown in formula (43).

$$(:\}1.1\{: \xrightarrow{M}:\}1.3\{: \xrightarrow{p1}:\}1.4\{: \xrightarrow{Sebastian}, (:\}1.1\{: \xrightarrow{M}:\}1.3\{: \xrightarrow{p1},:\}1.2\{: \xrightarrow{p10}) \Rightarrow:\}1.5\{: \xrightarrow{p1,p10}:\}1.6\{:$$

$$\xrightarrow{p1\ is\ p10's\ father}) \Rightarrow:\}1.7\{: \xrightarrow{Sebastian}[\text{Answer} = \text{Sebastian}] \quad (43)$$

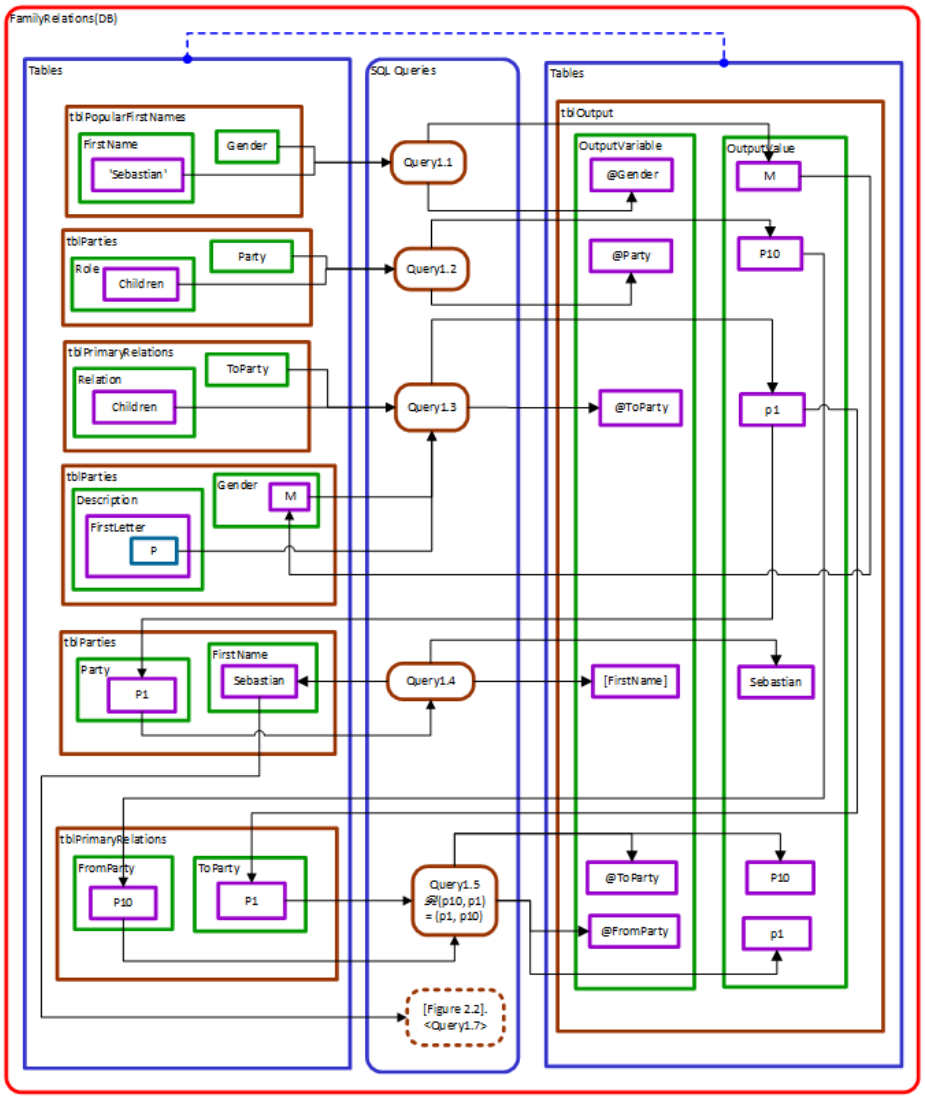

Figure 2. *Infobody Chart for Question 1(Part 1).*

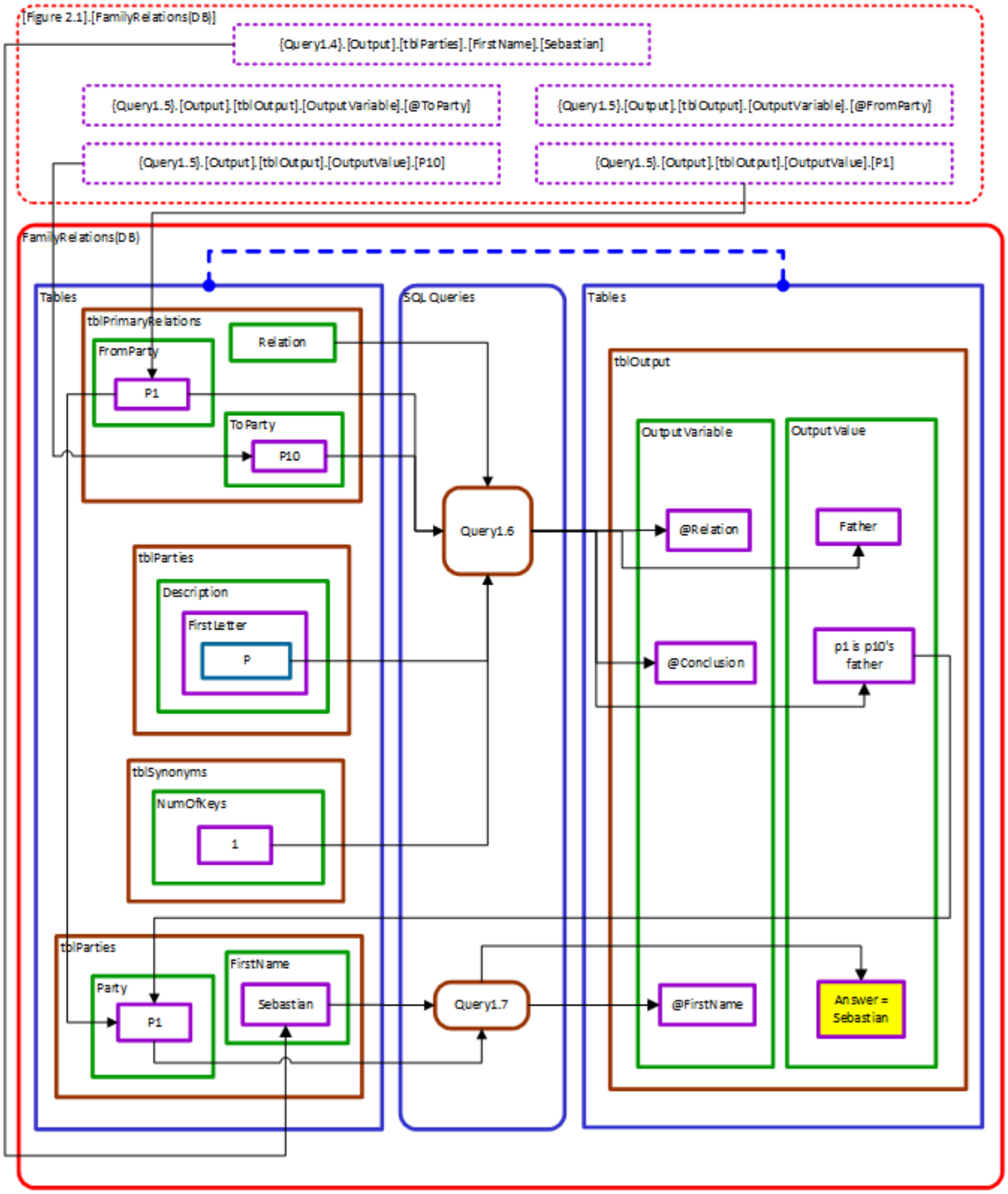

Figure 3 *Infobody Chart for Question 1 (Part 2).*

In Figures 2 and 3 we used some dashed lines for some infobodies which means those infobodies are in another chart. The blue dashed line means the two connected infobodies [Tables] (also in blue borders) are the same.

8.2. Question 2

Question 2 (Q2): What do you call your mother's brother's daughter?

SQL Code for Q2

```
-- For Question 2 (Q2): What do you call your mother's brother's daughter?

-- Query2.1: Find your mother's brother

declare @Relation varchar(50)
select @Relation = [Relation] from [tblDerivedRelations]
```

```
where [LeftRelation] = 'Brother' and [RightRelation] = 'Mother'
--print @Relation
-- Uncle
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query2.1', 1, '@Relation', @Relation)

----------
-- Query2.2: Find your mother's brother'daughter

-- get previous output
declare @PrevRelation varchar(50)
-- Get output value
select @PrevRelation = [OutputValue] from [dbo].[tblOutput]
where [Query] = 'Query2.1' and [StepNum] = 1
-- print @PrevRelation
-- Uncle

declare @Relation varchar(50)
select @Relation=[Relation] from [dbo].[tblDerivedRelations]
where [LeftRelation] = 'daughter' and [RightRelation] = @PrevRelation
print @Relation
-- Cousin

insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query2.2', 2, '@Relation', @Relation)

-- check output
select * from [dbo].[tblOutput] order by [Query], [StepNum]
```

The infobody chart for Q2 is in Figure 4 together with the chart for Q3 to show the common unit structures.

8.3. Question 3

Question 3: What do you call your mother's brother's father?

SQL Code for Q3

```
-- For Question 3 (Q3): What do you call your mother's brother's father?

-- Query3.1: Find your mother's brother

declare @Relation varchar(50)
select @Relation = [Relation] from [tblDerivedRelations]
where [LeftRelation] = 'Brother' and [RightRelation] = 'Mother'
print @Relation
-- Uncle
insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query3.1', 1, '@Relation', @Relation)

----------

-- Query3.2: Find your mother's brother's father

-- get previous output
declare @PrevRelation varchar(50)
-- Get output value
select @PrevRelation = [OutputValue] from [dbo].[tblOutput]
where [Query] = 'Query3.1' and [StepNum] = 1
-- print @PrevRelation
-- Uncle

declare @Relation varchar(50)
select @Relation=[Relation] from [dbo].[tblDerivedRelations]
where [LeftRelation] = 'Father' and [RightRelation] = @PrevRelation
print @Relation
-- Grandfather

insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query3.2', 2, '@Relation', @Relation)

-- check output
select * from [dbo].[tblOutput] order by [Query], [StepNum]
```

Infobody Charts for Q2 and Q3

The infobody charts for Q2 and Q3 are shown in Figure 4.

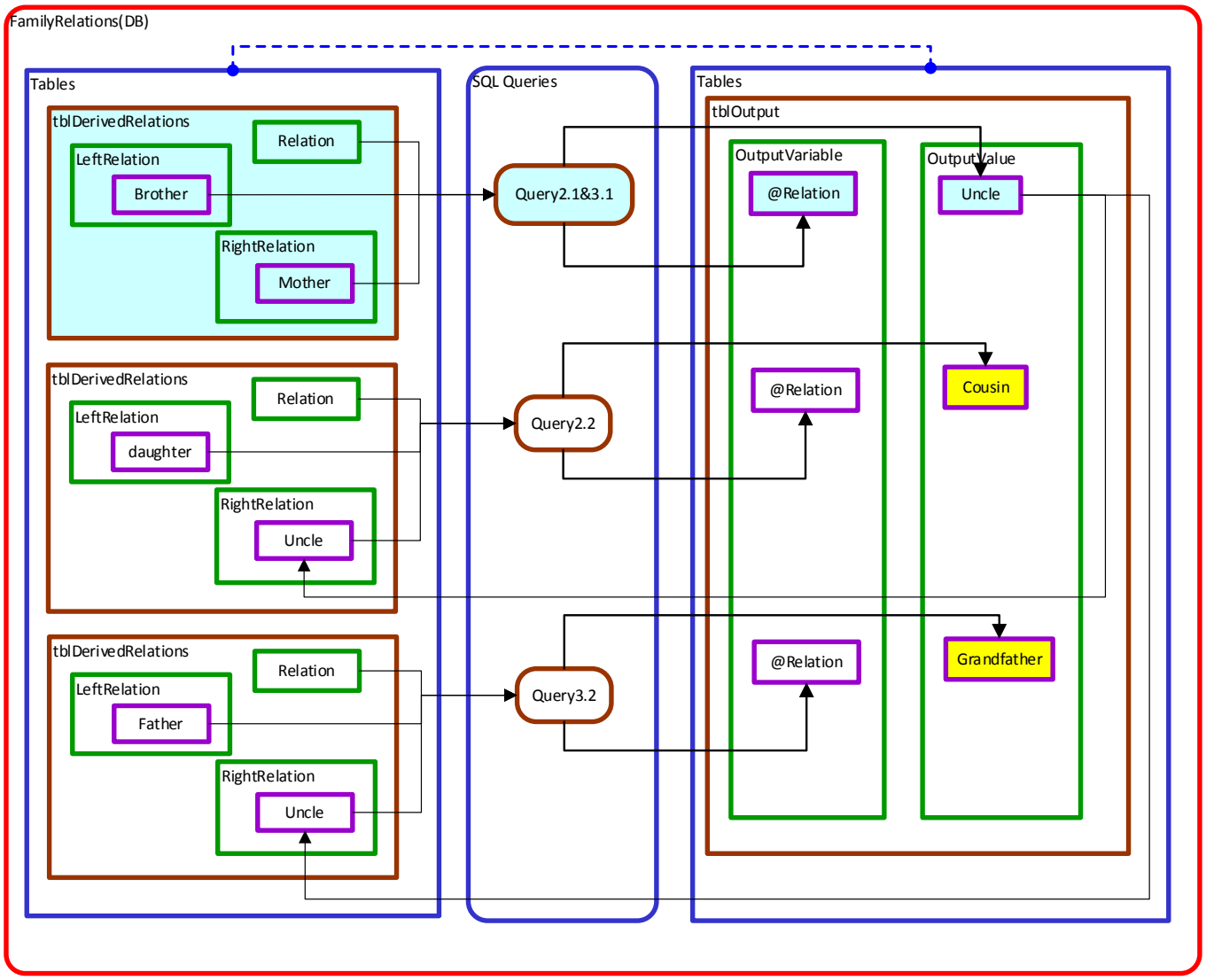

Figure 4 *Infobody Chart for Question 2 & 3*

In Figure 4, processor <Query2.1&3.1> is actually the same query for the two user questions. The corresponding unit structure is a shared unit structure colored in light blue in the figure.

The answer structure for Q2 is:

$$:\}\text{Query2.1\&3.1}\{: \xrightarrow{Uncle} :\}\text{Query2.2}\{: \xrightarrow{Cousin} [\text{Answer} = \text{Cousin}] \quad (44)$$

while the answer structure for Q3 is:

$$:\}\text{Query2.1\&3.1}\{: \xrightarrow{Uncle} :\}\text{Query3.2}\{: \xrightarrow{Grandfather} [\text{Answer} = \text{Grandfather}] \quad (45)$$

8.4. Question 4

Question 4 (Q4): (Missing Role Example) You have 3 sisters and you are the only one brother of them. What should your third sister's son call you ?

SQL Code for Q4

```
-- Q4: You have 3 sisters and you are the only one brother of them. What should your third sister's son call you ?

-- Query4.1:
-- 1) Add missing party daughter3 as p24
declare @MaxPid as int
select @MaxPid = max(pID) from [dbo].[tblParties]
--print @MaxPid
insert into [dbo].[tblParties]([Party], [Role], [Description], [Gender])
values ('p'+ cast(@MaxPid + 1 as varchar(10)), 'Daughter3', 'Person', 'F')

-----------------------
```

```
-- Query4.2
-- 2) Add daughter3's son as p25
declare @MaxPid as int
select @MaxPid = max(pID) from [dbo].[tblParties]
--print @MaxPid
insert into [dbo].[tblParties]([Party], [Role], [Description], [Gender])
values ('p'+ cast(@MaxPid + 1 as varchar(10)), 'Daughter3''s son', 'Person', 'M')

----------------------
-- Query4.3
-- 3) Add into relation [Mother]
declare @D3P as varchar(10)
select @D3P= [Party] from [dbo].[tblParties]
where [Role] = 'Daughter3'
print @D3P
-- p24

declare @D3Son as varchar(10)
select @D3Son = [Party] from [dbo].[tblParties]
where [Role] = 'Daughter3''s son'
print @D3Son
-- p25

-- add into Relation [Mother]
insert into [dbo].[tblPrimaryRelations]([Relation], [FromParty], [ToParty], [Description])
values('Mother', @D3P, @D3Son, @D3P + ' is ' + @D3Son + '''s mother')

----------------------
-- Query4.4
-- 4) Add into relation [Son]
declare @D3P as varchar(10)
select @D3P= [Party] from [dbo].[tblParties]
where [Role] = 'Daughter3'
print @D3P

declare @D3Son as varchar(10)
select @D3Son = [Party] from [dbo].[tblParties]
where [Role] = 'Daughter3''s son'
print @D3Son

-- add into Relation [Mother]
insert into [dbo].[tblPrimaryRelations]([Relation], [FromParty], [ToParty], [Description])
values('Son', @D3Son, @D3P, @D3Son + ' is ' + @D3P + '''s son')

--------------------------------
-- Query4.5
-- 5) Get [Mother] for (p24, p25)
declare @D3P as varchar(10)
select @D3P= [Party] from [dbo].[tblParties]
where [Role] = 'Daughter3'
print @D3P
--p24

declare @D3Son as varchar(10)
select @D3Son = [Party] from [dbo].[tblParties]
where [Role] = 'Daughter3''s son'
print @D3Son
--p25

declare @Relation varchar(50)
select @Relation = [Relation] from [dbo].[tblPrimaryRelations]
where [FromParty] = @D3P and [ToParty] = @D3Son
print @Relation
-- Mother

insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query4.6', 6, '@Relation', @Relation)

---------------
-- Query4.6
-- 7) Get [Brother] ~ [Mother] = [Uncle]
declare @Derived varchar(50)
select @Derived = [Relation] from [dbo].[tblDerivedRelations]
where [LeftRelation] = 'Brother' and [RightRelation] = 'Mother'
print @Derived
--Uncle

insert into tblOutput([Query], [StepNum], [OutputVariable], [OutputValue]) values('Query4.7', 7, '@Derived', @Derived)

-- check output
select * from [dbo].[tblOutput] order by [Query], [StepNum]
```

Infobody Chart for Q4:

The infobody chart for Q4 is shown in Figure 5.

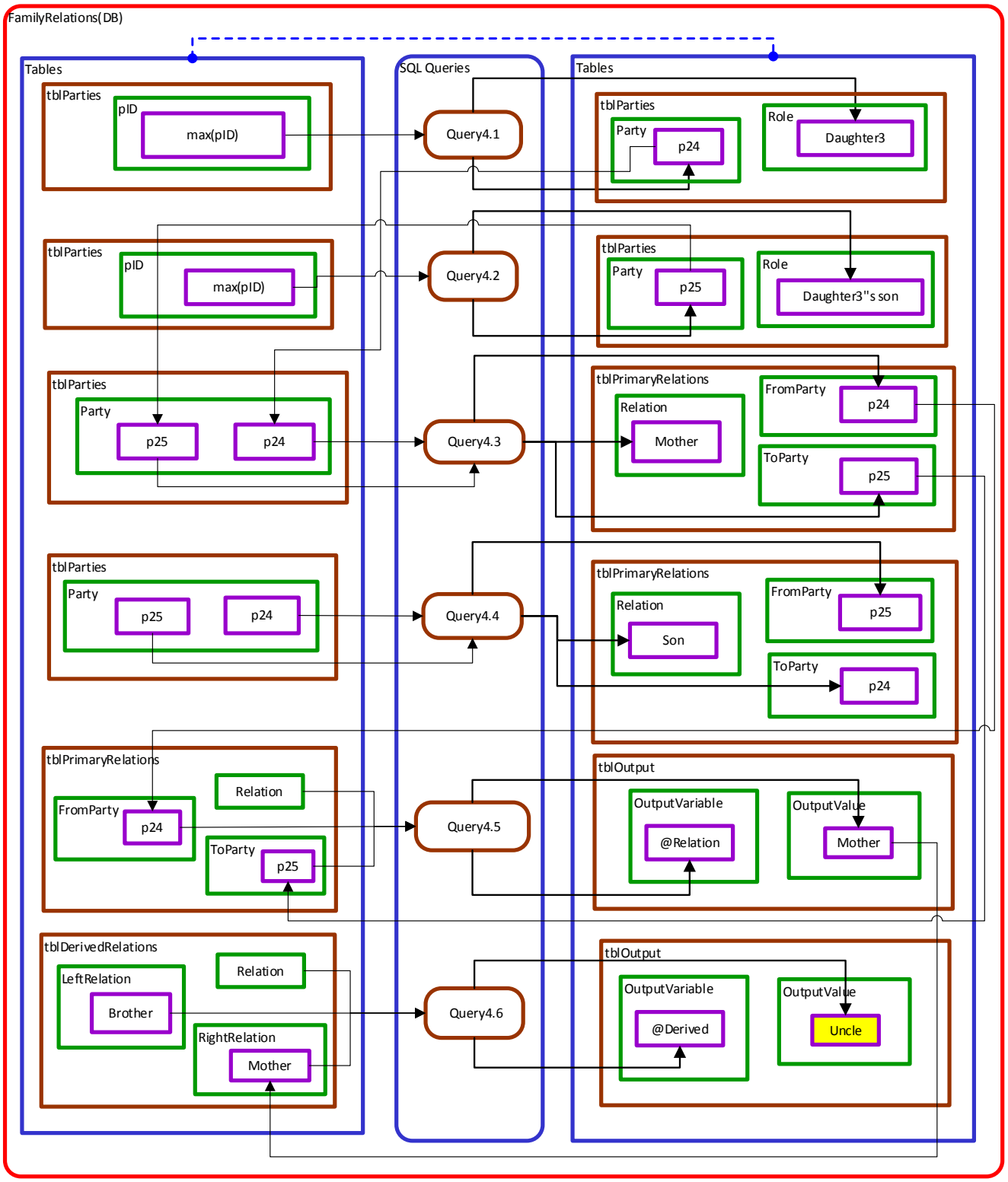

Figure 5. *Infobody Chart for Question 4*

The infobody expression for Q4 is shown as follows.

$$(:\}4.1\{: \xrightarrow{p24},:\}4.2\{: \xrightarrow{p25}) \Rightarrow :\}4.3\{: \xrightarrow{p24,p25} :\}4.5\{: \xrightarrow{Mother} :\}4.6\{: \xrightarrow{Uncle} \text{[Answer = Uncle]} \quad (46)$$

Note, the unit structure:}4.4{: is not showing up in the answer structure (46) because it is for inserting p25 and p24 into tblPrimaryRelations for relation [Son] which is the reverse of relation [Mother]. Usually when we add a pair of new parties we need to add them into tblPrimaryRelations for both relations (one relation and the reverse relation) even if one of

the newly added relations may not be used to answer this specific question.

9. Conclusions and Prospects

This paper is the first attempt to apply Infobody Theory and Infobody Model to Logical Artificial Intelligence (LAI). It seems successful and we can conclude that Infobody Theory and Infobody Model is a useful tool for LAI.

The current AI is based on Deep Learning and Backpropagation which is based on big data and we can refer it as Data AI or DAI. As we all understand, human brain has two major functions: memory and thinking. DAI is trying to simulate the memory function and LAI is trying to simulate the thinking function.

Infobody theory can be applied to LAI because it focuses on all kinds of infobody structures and two kinds of infobody structures are specifically suitable to LAI which are thinking structures and knowledge structures. Thinking structures focus on human thinking such as the family relations structure based on common sense. Knowledge structures focus on existing academic knowledge in each specific academic area based on academic publications such as academic books and published papers.

Infobody (IB) model is primarily for implementing infobody structure studies in a relational database based on graph theory on containing chains and the relations between input infobodies, processors and output infobodies. It can be applied to many areas such as computer architectural analysis and design, management system analysis and design and etc. Logical AI is also a good area for IB application. One interesting topic for thinking structures is to simulate creative thinking structures which may empower LAI in creative thinking.

This paper used the thinking structure for family relations as an example to show how LAI should work and how to apply IB approaches for LAI studies. The family relations database is a basic database to present the basic logic between family relations, but the tables may not be complete and some missing data may be added into the tables. In most cases for LAI studies using one or more databases to present the basic logic in a specific area is a good idea but it may not be necessary, which means IB modeling for LAI may not need a basic logical database and the specific logic may be implemented in the IB model directly. The IB model implementation itself is always based on a database with two primary tables, one for containing chains and one for Input-Processor-Output (I-P-O) relations (e.g. unit structures for LAI). Then various calculations (such as answer structures for LAI) can be defined and implemented in the IB Model.

Usually, LAI needs variety of basic databases to present the basic logic in each specific area including social (e.g. Family Relations) and academic (e. g. Differential Geometry, Theoretical Physics, Molecular Biology, and etc.). Those databases need scientists and IT professionals to work together and both of them need to have basic understanding of Infobody Theory and Infobody Model.

In this paper we used Infobody charts to demonstrate the IB model. The real IB model for Family Relations will be discussed in details in another paper [Infobody Model Implementation for Logical AI].

Usually, the basic logical database and the IB model database should be developed by authority organizations with lower infobody structure entropy values ([7] Chapter 4: Chaos and Entropy in Infobody Structures) and published in the cloud for public accesses.

As we all know, the mature DAI is based on huge amount of data published on the web sites with the progressive web technologies over last few decades. We can also imagine the mature LAI will need a lot of efforts from academic professors, IT professionals and infobody professionals working together to build a lot of IB models on the clouds for a few decades. Suppose all academic areas are implemented in knowledge structures with IB models in clouds, and all commonsense areas such as family relations are implemented in thinking structures with IB models in clouds, then, any LAI app, like ChatGPT, should be able to query some of them to answer any logical questions. Also, it is possible to make those IB models for LAI available for all kinds of robots to simulate creative thinking.

LAI and DAI can be combined to build Total Artificial Intelligence (TAI). The way to combine LAI and DAI is to use DAI to estimate the parameters in each processor in the IB Model for LAI. Also, some variables may be setup for specific properties such as someone's first name, someone's gender and etc. in the family relations example.

One thing we did not implement in this paper is how to find the answer structure based on the user question in a language. We will need more studies in this area with the help of NLP [13], but the current database technologies with huge storage capacity in clouds already provided a basic solution for this issue, that is, list all answer structures with the corresponding questions in the language, even with all possible synonyms and sentence variations.

Abbreviations

LAI	Logical Artificial Intelligence
DAI	Data Artificial Intelligence
TAI	Total Artificial Intelligence
IB	Infobody
I-P-O	Input-Processor-Output

Author Contributions

Yuhu Che is the sole author. The author read and approved the final manuscript.

Conflicts of Interest

The author declares no conflicts of interest.

References

[1] Eka Roivainen (2023): I Gave ChatGPT an IQ Test. Here's What I Discovered. Scientific American. https://www.scientificamerican.com/article/i-gave-chatgpt-an-iq-test-heres-what-i-discovered/

[2] Anas Al-Masri (2024): How Does Backpropagation in a Neural Network Work https://builtin.com/machine-learning/backpropagation-neural-network

[3] Jack Minker (2000): Logic-Based Artificial Intelligence https://link.springer.com/book/10.1007/978-1-4615-1567-8

[4] Kory Becker (2018): Logical-Based Artificial Intelligence and Expert Systems https://www.primaryobjects.com/2018/07/23/logical-based-artificial-intelligence-and-expert-systems/

[5] John McCarthy (1999): CONCEPTS OF LOGICAL AI https://www-formal.stanford.edu/jmc/concepts-ai/concepts-ai.html

[6] Cogni Down Under (2024): Inside Logical AI: Architectures, Methods, and Metrics Pushing Towards More Explainable Reasoning https://medium.com/@cognidownunder/inside-logical-ai-architectures-methods-and-metrics-pushing-towards-more-explainable-reasoning-bc1d526072cc

[7] Yuhu Che (2022): Infobody Theory and Infobody Model, Page Publishing (2022), ISBN 978-1-6624-5074-7. https://www.amazon.com/Infobody-Theory-Model-Yuhu-Che/dp/1662450745

[8] Phiroze Hansotia (2003): A Neurologist Looks at Mind and Brain: "The Enchanted Loom" https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1069062/

[9] Merriam-Webster Dictionary (2023) https://www.merriam-webster.com/dictionary/information

[10] Relation theory: https://oeis.org/wiki/Relation_theory

[11] Power Set (2023): https://byjus.com/maths/power-set/#:~:text=What%20is%20a%20Power%20Set,set%2C%20of%20a%20given%20set

[12] Most Popular Male and Female First Names: https://www.dia.govt.nz/diawebsite.nsf/wpg_URL/Services-Births-Deaths-and-Marriages-Most-Popular-Male-and-Female-First-Names?OpenDocument

[13] Complete Guide to Natural Language Processing (NLP) – with Practical Examples https://www.machinelearningplus.com/nlp/natural-language-processing-guide/