Alternative Technologies™

Logic for Serious Database Folks Series

by David McGoveran, Alternative Technologies

LEGAL NOTICE AND LICENSE

IMPORTANT CONTENT NOTICE

It is anticipated that many readers of this document will have some familiarity with the writings of other authors, especially but not limited to E. F. Codd ("EFC"), C. J. Date ("CJD"), and H. Darwen ("HD"), or familiarity with Date and Darwen's The Third Manifesto ("TTM") and related writings. Material in this series will not comply with all of TTM and definitions therein, nor with any other interpretation of the work of EFC. Readers should not assume otherwise of an independent work. I will, from time to time, refer to specific passages in the works of EFC, CJD and HD for purposes of commentary or to illustrate some issue. When I disagree with a some construct by EFC, CJD, or HD and am aware of it, I will say so and give a justification.

CONTACT BY REVIEWERS

Reviewers are encouraged to convey their comments and criticisms to me directly via email addressed to **mcgoveran@AlternativeTech.com**., and including "REVIEW" in the subject line. Please provide document title, revision date, and page when referencing any passage. I will do my best to respond in a timely manner and will try to answer specific questions if there is a reasonably short answer.

Chapter 3: The Deductive Subsystem
Logic for Serious Database Folks Series

by David McGoveran, Alternative Technologies

Logic is using your wits, and the proof is at your wit's end.

I. INTRODUCTION

Now that we have some concepts and vocabulary (such as a meta-language) in hand, let's examine the components and defining properties of *formal systems*, and more particularly, *formal logical systems*. A formal logical system is a special type of formal system, one that has a special type of language that permits deductive reasoning using rules of inference.

Before getting into the details, we present the reader with a somewhat informal warning. It is tempting to think of a formal logical system as one from which one can compute truth or falsity symbolically. Technically, this is incorrect. A formal logical system is an abstract system for reasoning by mechanical procedures or rules that constitute its abstract structure (a.k.a., grammar or syntax). These mechanical aspects of the system are its *syntax*. Until formally interpreted, a formal logical system is devoid of *interpretive semantics* (i.e., meaning) even though it will usually suggest or have associated with it some intended use.[1] However, a formal logical system does contain *semantic* components that permit it to be interpreted in terms of a second system. Because the system is abstract, it cannot be used (outside of investigating its intrinsic properties) without identifying a second system to which it is to be applied. That second system must have elements – including objects, relationships, and operations – that correspond to the symbols of the formal logical system so that a morphism or mapping between the two systems may be asserted.

[1] Such use is sometimes said to be the system's intended or *canonical semantics*, and is usually rather abstract… but more on this later.

II. SOME CATEGORIES OF FORMAL SYSTEMS

Formal systems may be classified into categories. The terms used to identify these categories vary in the literature. The terms used here are convenient for our purposes and, hopefully, descriptive.

Among the categories of formal systems are the following:

- **Formal Language Systems (Formal Languages)** – Every formal system includes a formal language, consisting of a vocabulary divided into a set of classes, and a set of *grammatical rules*. Formal languages are often studied in their own right. The vocabulary of a formal language consists of *terms* (similar to words in a natural language), each of which belongs to a class. This is similar to the division of a natural language vocabulary into classes such as nouns, verbs, adjectives, and the like. The grammatical rules govern how terms may be combined into formal *expressions*. Grammatical rules often specify the order or contiguity of terms according to the classes to which those terms belong. This is similar to English grammatical rules that, for example, permit a noun to be followed by a verb in a declarative sentence.

    - **Symbolic Languages** – Some formal languages are symbolic languages. In a symbolic language, the terms of the *vocabulary* are either simple or compound symbols. The collection of simple symbols is called an *alphabet*. Compound symbols are built up from other symbols according to *composition rules* (these are not always specified)[2]. Terms are then combined into symbolic language expressions, called *formulas*, according to *formation rules*. A formula that complies with the languages formation rules is said to be a *well-formed formula*, usually abbreviated as "*wff*"[3]. As with the terms of any formal language, symbols are typically divided into classes, and the formation rules specify how to combine terms according to the classes to which those terms belong.

    - **Deductive Languages** – In addition to the grammatical rules (or formation rules in the case of a symbolic language), a deductive language incorporates a set of inference rules. ***Inference rules*** are either *deductive rules* or *rewrite rules* that are intended to preserve some property of expressions in the deductive language. Usually, the relevant property is some notion of truth or validity. For example, given one or more expressions that are deemed "true" by some formal criteria, applying an inference rule to those expressions generates a new expression that will also be "true" by those same criteria. One says that the new expression has been "deduced" or "inferred" from the old expression. (If the

[2] Deductive languages often have no vocabulary other than the alphabet, so that its symbols are used as primitive or primary terms.
[3] Wff is preferably pronounced "wiff".

language is also symbolic, we would use "formula" in place of "expression" above.)

- **Uninterpreted Formal Systems** – Unless otherwise specified or obvious from context, a formal system is *uninterpreted*, by which we mean that its symbols, terms, and expressions or formulae are abstract. The components of an uninterpreted formal system have no specific meaning *per se*. That said it is important to be aware that the formal system may take on a certain rather abstract meaning by virtue of the syntactic or structural relationships among the formal system's components. An uninterpreted formal system has, at best, such an *abstract semantics*. We return to this notion of abstract semantics in the discussion of *intended* (a.k.a. canonical or standard) *interpretations* below. We call the general structure of a formal system that does not involve interpretation the *Deduction Subsystem*. We will study it below in the context of *uninterpreted formal* <u>*logical*</u> *systems*.

- **Interpreted Formal Systems** – A formal system that is understood as representing some specific subject is said to be interpreted by that subject, thereby giving the components of a formal system *meaning* – an *interpretive semantics*. The components (symbols, terms, and expressions or formulae) of a formal system are provided *formal interpretations* by relating each primitive component of the formal system to some primitive components of the subject via *rules of correspondence*. (Again, by *primitive*, we mean "treated as fundamental" or "unanalyzable".) A ***rule of correspondence*** is a binary relation between a primitive component of the formal system and a primitive component of the subject. Each symbol, term, or expression then acts as a name (in the formal system) for some component of the subject, said to be the name's denotation (in the subject). If every symbol, term, and expression of the formal system is given an interpretation, the formal system is said to be a *fully interpreted* (often, just "interpreted") *formal system*. If any component of the formal system is not interpreted, the formal system is said to be *partially interpreted*. We call the general structure of a formal system that enables its interpretation the *Interpretation Subsystem.* We will study the *Interpretation Subsystem* in the context of *interpreted formal logical systems* in some detail in the next article. As we shall see, any limiting properties of an uninterpreted formal system are not mitigated by its interpretation.

    - **Formal Representation Systems** – The simplest type of the interpreted formal systems is a *formal representation system*, consisting of a formal language and an Interpretation Subsystem. The Interpretation Subsystem is used to give the components of the formal language meaning. In this way, the formal language may be said to represent the subject. Formal representation systems are primarily descriptive in nature. They do not include any deductive or inferential capability.

- **Formal Logical Systems** – The language of formal logical systems is a deductive language as defined above. In addition to purely structural properties, the formal system has deductive properties as well. The Deductive Subsystem is used to give the formal system deductive power. The deductive languages of almost all formal logical systems are also symbolic languages. Formal logical systems may be interpreted, partially interpreted, or uninterpreted.

- **Formal Axiomatic System** – A formal system that is developed according to the axiomatic method is called a formal axiomatic system. By requiring definitions and propositions to be supported by an established set of axioms and primitive terms, the axiomatic method avoids the possibility of *infinite analytical regress.*[4] (Nonetheless, an axiomatic system can permit self-reference if it is not also *constructive*[5].) Usually, the formal system is initially defined by a set of propositions that are held to be valid, and a set of axioms and inference rules is then developed by which that set of propositions may be proven true given that the set of axioms are true. The set of axioms may have certain important properties such as *minimality* or *independence* (both defined in a subsequent article) or not.

- **Constructive Formal System** – A formal system is constructive if every component is defined from primitives using only an *effective procedure* (a finitely specified procedure that yields a definite result after a finite number of finite mechanically-applied operations). This requirement applies to both the syntax of the system as given by the Deductive Subsystem and to the *interpretive semantics* (if any) of the system as given by the Interpretive Subsystem. Every aspect of a constructive formal system is expressible in terms of a finite set of operations on primitives, forming a hierarchy of concepts and structures with a definite base. Note that circular definitions are impossible: an object of the system only exists if one constructs it from a primitive or a previously constructed object. One must be careful when discussing specific constructive formal systems, as these often involve additional requirements not intended by our general definition. Examples include most axiomatic set theories classified as constructive.

III. FORMAL LOGICAL SYSTEMS

The study of formal logical systems is divided into two parts which must be kept distinct, both conceptually and in practice. These disciplines are known as *proof theory* and *model theory*, roughly corresponding to the study of *syntax* and *semantics*, respectively. Many misunderstandings and errors pertaining to formal logical systems, their strengths, weaknesses, appropriate uses, and inappropriate uses arise because the two disciplines and their corresponding concepts become conflated. These two disciplines may be formally related in very specific ways, especially as regards the properties of interpreted formal logical systems. However, proof theory pertains to purely syntactic properties while model theory pertains to

[4] The process of giving definitions by analyzing each term into component terms and then each of those into its component terms is called an *infinite analytical regress*. It is not an effective procedure because any rule for termination is necessarily arbitrary. Such a process is called an *infinite analytical regress*.
[5] Definition by analysis is to be contrasted with *constructing* terms from a finite set of primitive terms. By ***constructive***, we mean that definitions of primitive terms are concrete rather than abstract, and one must give an effective procedure for constructing derived terms from primitive terms.

semantic properties. In the material that follows, the reader is encouraged to think carefully about why these two disciplines are necessarily distinct in formal systems.

Given the two disciplines, it is useful to divide the structural components of formal logical systems into two portions – Deduction Subsystem and Interpretation Subsystem, corresponding to the subject matter of proof theory and model theory respectively.

In the remainder of this article, we will discuss the Deduction Subsystem. The issues and concepts introduced above are examined a bit more formally below. We will reserve further discussion of the Interpretation Subsystem in the next article of this series.

Formal Logical Systems involve two disciplines: Proof Theory and Model Theory. Their objects of study must never be conflated.

Proof Theory

The discipline called ***proof theory*** is the study of an uninterpreted formal logical system as a purely abstract entity. We say that a formal logical system forms a ***theory***, and the language of the theory is called the ***object language***. The object language is used to express reasoning about the abstract subject matter of the theory (i.e., the theory's *object*), a process usually called *deduction* or *inference*. More often than not, the object language will be an ***algebra***, defined as a language for expressing and reasoning about relationships among its terms.

By sharp contrast with the object language, we reason about and talk about a formal logical system using a different logic and language called the ***meta-language*** (sometimes called the ***observer's language***). It is sometimes said that when we want to discuss a theory, we have to do so from within another formal logical system – a theory about a theory – formally called a ***meta-theory*** because it is "*meta to*" or "*outside of*" the object of study. One can then see that it is the object language of this meta-theory that is then called a *meta-language*. Within informal discussion of a formal system, many different meta-languages may be used, any of which may be unspecified and so is an *informal meta-language*. Often, logic texts use ordinary English at the informal meta-language for the study of multiple formal logical systems, and never make this designation explicit.

Any language used to discuss a system, including that used in this series, is serving as a meta-language. When we want to prove that a formal logical system has certain properties or to discuss its alleged properties, we do so in the meta-language. When we engage in formal analysis of formal systems, we must select and use a *formal meta-language* – a language with a formal specification. The reasons for this dictum will become clear when we explore the properties that differentiate formal systems in a later article[6]. We observe that a meta-language needs to be sufficiently rich (in vocabulary and syntax) to express the concepts of the formal system it is being used to discuss or analyze faithfully. That is, the meta-language must not import extraneous concepts, operations, or relationships into the formal system under study, and must not alter or restrict any of that formal system's intrinsic concepts, operations, or relationships. We say that the meta-language must be at least as ***expressively powerful*** as the object language. Expressive power is an important syntactic property of languages, and one we will encounter again and again.

A second important observation is that the object language and the meta-language may or may not be the "same" in terms of their formal language specifications. Even if the specifications of these languages are same, an object language and a meta-language are two *separate, distinct uses* of that language. When the same formal language is used for both object language and meta-language, it is important to distinguish between those distinct uses, being constantly aware of and acknowledging which one is being used in the course of any given analysis.[7]

A Logical System's Object Language and Meta-Language may or may not be formally equivalent, but their uses must never be conflated.

In a formal logical system, the composition syntax of the object language is sometimes given as a set of rules called *formation rules*. ***Formation rules*** govern how the symbols of vocabulary are combined to form terms and how terms may in turn be combined to create formulas. The remainder of the syntax is given by *axioms* and *inference rules*. ***Axioms*** are formulas that are presumed *a priori* to be true. ***Inference rules*** are used to deduce new formulas from existing formulae (such as axioms), while preserving some special property.

Model Theory

The second discipline, called ***model theory***, is the study of interpretive semantics. An application of model theory comprises a set of *rules of correspondence*, a set of *truth values*, a set of *truth assignments*, an *evaluation language*, an *evaluation procedure*, and a ***subject system*** (either formal or informal) having elements and relationships among (or operations on) those elements. A subject system to which a formal logical system (or deductive theory) may be applied (successfully) is said to be a ***model*** of the theory. A subject system typically has an associated *subject language* in which knowledge of the subject system is normally expressed, independent of other system. Model theory examines semantic elements (meaning and truth) of a *subject*

[6] Currently, the topic of differentiating formal logical systems by their properties is planned for Chapter 5.
[7] As we will see later on, it is a special challenge to determine whether or not certain properties of a formal logical system can be proven in the meta-language if the meta-language is the same as the object language.

language (about the *model*) by means of syntactic elements (formulas and proofs) of a corresponding object language (in the deductive theory). Put another way, the objects of study of *model theory* are possible models of theories in a formal language.

IV. THE DEDUCTION SUBSYSTEM

We begin our study of the Deduction Subsystem with some definitions, which will be used extensively as we go on. For pedagogical purposes, we will consider a formal logical system as comprising of types of objects (see Figure 3.1): an Alphabet, Composition Rules, Vocabulary, Formation Rules, Definition Rules, Axioms, and Rules of Inference.[8] These definitions are general and applicable to all logical systems, being refined or narrowed as appropriate to the specific logical system being discussed. Note that definitions are given in meta-language: the entirety of definitions do not belong to the object language *per se*.[9]

Alphabet

In general, a formal logical system will have an ***alphabet***[10] consisting of a set of *symbols*. As discussed earlier (see On Language), a symbol is a kind of sign: A symbol denotes meaning other than itself. The symbols of a formal language are often classified according to their intended use.

Composition Rules

A group of symbols combined according to rules of composition and having a denotation is called a *term*. Very often a formal logical system will use symbols for the primitive terms of its vocabulary so that, in effect, the only composition rule is one that makes every symbol a primitive term. For a formal logical system, this simplification usually does not result in any loss of expressive power. However, it may be disadvantageous in a formal representation system, where the structure of terms may have explanatory power. As an example, consider a formal representation system for the phonetics of a natural language. Clearly, in such a case, we would want to formalize the rules by which phonemes are combined to form words and so an alphabet of symbols denoting phonemes and *composition rules* for creating words from those phonemes (denoted by terms of a vocabulary) would be advantageous.

[8] Other authors may divide a formal logical system into a slightly different set of components, or use different names for them. We prefer this one for pedagogical reasons.
[9] Church (1944) refers to the specific meta-language used for this purpose as the "syntax language".
[10] The term alphabet arises from its proper use in formal language theory.

Alphabet:
A Set of Symbols

Composition Rules:
A set of rules for creating *terms* from the *symbols* of the Alphabet.

Vocabulary:
A Set of Logical Terms: Grouping indicators: "(" ")" and Operators (Connectives)
A Set of Non-Logical Terms: Constants and Variables serving as abstract placeholders (operands of operators).

Formation Rules:
A set of rules for creating well-formed (i.e., syntactically correct) formulae from the terms vocabulary.

Definition Rules:
A set of rules for defining new terms from existing terms.

Axioms:
A set of well-formed formulae, given as having validity (or some other property) *a priori*.

Rules of Inference:
A Set of Deductive Rules having the form: Given that <set of wffs > satisfies <specific property>, therefore <wff> satisfies <specific property>. It is the form of wffs that determine applicability, not the particular symbols used to express them. By definition, axioms are provable from an empty set of premises.
A Set of Rewrite Rules that are restricted rules of term substitution of the form: Given <wff containing term>, therefore <wff with substitute term replacing term>.

FIGURE 3.1. Deduction Subsystem Components

Vocabulary

A ***vocabulary*** is a collection of terms organized into categories according to their intended use. Typically, terms are classified into variables, grouping indicators, and operators (connectives). In a formal logical system, the primitive terms (possibly singular symbols) of a vocabulary are further characterized as either logical or non-logical, leading to a natural division of the vocabulary (explained below). Two or more terms may be combined in such a way that the resulting combination is denotative (i.e., acts as a name of some denotation). In such cases, that combination may itself be referred to as a term. Two or more terms are usually combined with some selection of symbols for grouping indicators and connectives to create more complex terms called *derived terms*. In addition, terms concatenated according to Formation Rules (see below) create a special kind of term called a *formula*.

When logicians use terms like “symbol”, “alphabet”, “vocabulary”, “term”, “word”, and “sentence”, they are usually talking about the Vocabulary of the formal logical system, bypassing the use of an Alphabet and any Composition Rules. As discussed above, the terms of a symbolic language are often singular symbols, so that Alphabet and Vocabulary become synonymous. In the remainder of this text, we will assume this simplification. Furthermore, we will refer to primitive terms as symbols, inasmuch as any of a formal system’s primitive terms, by definition, have no component that is denotative.

<u>*Logical Symbols*</u>

Logical symbols[11] are those that have an abstract meaning that is independent of any meaning assigned to them by a particular interpretation or application. Logical symbols include, assuming the system incorporates them, ***n***-ary operators[12] (including binary operators called *connectives*), grouping indicators, and binding indicators (such as the quantifiers, lambda symbol, etc.). More often than not, logical operators are either monadic or dyadic. When used with uninterpreted non-logical symbols, they are used to form the formulae[13] and sequences of formulae studied in proof theory.

By definition, a logical ***n***-ary operator operates on ***n*** operands. Each of those operands is equivalent, either by assignment or under an interpretation, to a one of the system’s ***m*** *truth value constants* (see below). For every combination of truth value constants assigned to those operands, the operator “expression” is then equal to one of the system’s ***m*** truth value constants. There are many ways in which this can be done. Specifically, there are $m{\uparrow}m^n$ (the symbol “↑” here means exponentiation) possible ***n***-ary operators for which the operands can take ***m*** values. Not all of these operators will be independent in the sense that none can be obtained from the others by functional composition. A set of independent operators from which all others can be obtained by composition is called a ***basis***.

For example, suppose the formal logical system has exactly two truth value constants. Then, there will be $2{\uparrow}2^n$ ***n***-ary operators. If these operators are dyadic, there are sixteen (16) dyadic

[11] Church (1944) refers to these as "improper symbols." See footnote on "non-logical symbols" below.
[12] Most writers use the term “connectives” apparently because it is familiar from propositional logic. It improperly suggests only dyadic operators, excluding the common monadic operator of negation and failing to anticipate functional notations that permit the n-adic operators found in certain formal systems.
[13] Logicians frequently refer to formulae as “sentences”, but this term has specialized meaning in some formal logical systems and so we will avoid it. We will also avoid “expression” a term used for programming languages.

operators. Note that the number of operators increases very rapidly with number of truth value constants. For example, if number of truth value constants increases to three, then the number of dyadic operators becomes **3↑3*^{2}*** eighty-one (81).

Operators have a special double definition in a formal logical system, only one of which properly concerns us in the Deduction Subsystem. Their primary or syntactic definition is often given in terms of corresponding set theoretic operators (using a set theoretic meta-language) and (or[14]) Rules of Inference. The syntactic definition governs their deductive behavior and belongs to proof theory. It may be said to be an algebraic definition. With the exception of a requirement of consistency with the system's intended interpretation (discussed below), the meaning of logical symbols is controlled entirely by their syntactic definition.

Non Logical Symbols

The set of ***non-logical symbols***[15] includes, depending on the specific logical system, truth value constants, individual constants, variables, functions, and predicates. Non-logical symbols serve as placeholders in a formula of the uninterpreted theory such as the formulae that appear in a deduction. Such symbols are used in accordance with the Formation Rules (below), but otherwise merely reserve position for semantic elements – symbols from some interpretation. Any two symbols are assumed to have distinct denotations unless their denotations are explicitly asserted to be the same or proven to be the same. Non-logical symbols themselves convey no other information.

It follows that, as long as distinctions are preserved, any symbol not yet used can replace any symbol currently used as long as that replacement is done uniformly. In fact, if such replacement results in perceived a loss of comprehension, then you have failed to maintain adequate separation of syntax and semantics, of theory and interpretation. Excepting the system's intended interpretation (discussed under "Abstract Semantics" below), non-logical symbols have meaning only when one is assigned to them by an interpretation (discussed under Interpretation Subsystem below). Once a symbol is assigned additional information, it becomes part of a model.

Among the constants belonging to the non-logical symbols are symbols for a set of *abstract categoricals*, the members of which correspond to the set of truth values the formal logical system is intended to address. We refer to these symbols ***truth value constants***. The number of distinct truth value constants ***n*** is an important part of the definition of the formal logical system. A formal logical system with ***n*** distinct truth value constants is often referred to as "***n-valued***" and, if ***n*** is greater than two, "***multi-valued***" or "***many-valued***".

[14] Certain systems attempt to define an operator exclusively through the Rules of Inference that apply to it; that is, through how it is used.
[15] Church (1944) refers to these as "proper symbols" in the sense that they are used for proper names and variables.

For example, in a two-valued logical system the truth value constants are often given as "TRUE" and "FALSE". Although various symbols may be used, they invariably correspond to the meta-linguistic concepts of "*true*" and "*false*", respectively. Under any interpretation, the appearance of truth value constants TRUE and FALSE in an object language formula have a fixed replacement in the corresponding *evaluation language* (part of the Interpretation Subsystem) expression as the truth values /T/ and /F/, respectively (see below).

"TRUE" and "FALSE" are *truth value constants* in the object language.

Throughout this series I will use TRUE and FALSE (or similar uppercase symbols) for truth value constants in the object language. Similarly, I will use /T/ and /F/ (or similar uppercase symbols set off by forward slashes) for truth values in the model's *evaluation language*. (The *evaluation language* is discussed in the next article.)

Within proof theory, one does not evaluate the truth or falsity of a wff under an interpretation. Rather, an operator's corresponding truth value may be considered for all possible assignments of a truth value constant to each of the operands. For example, we will often want to know whether some wff of a formal logical system is or is not a *tautology*. This requires considering all possible assignments for each non-logical symbol and the corresponding truth value constant for each operand of each logical operator in the wff. Note that this process bypasses any assignment of meaning to individual constants and variables so that no specific interpretation of the non-logical symbols is involved. However, the assignment of abstract categoricals in the form of truth value constants has the effect of considering all possible interpretations of individual constants and variables.

Formation Rules

Within the Deduction Subsystem, an expression is usually called a ***formula***. A formula is a concatenation of terms (usually symbols) from the vocabulary. When that concatenation is in conformance with a prescribed collection of ***formation rules***[16] the result is said to be a "***well-formed formula***". The phrase well-formed formula is usually abbreviated "***wff***" and is pronounced "wiff". Any symbol from the group that is used to represent variables is typically considered a primitive wff. The most primitive multi-symbol components of a wff are usually called "*terms*". A sequence of symbols may have the appearance of a formula without being well-formed (it is "ill-formed"), meaning that it cannot be produced from the vocabulary by successive application of the formation rules. The obvious question regarding a set of formation

[16] A set of formation rules is sometimes called a grammar or syntactic rules, as in formal language theory.

rules is whether or not there exists some *algorithm* (a.k.a., an *effective procedure*) by which an arbitrary formula can be determined to be well-formed.

Definition Rules

Technically, all definitions should be given in a form that follows some set of explicitly stated "Definition Rules." This is particularly important with respect to *derived terms*, which in any way extend the expressiveness of the formal language. For most formal systems, Definition Rules are seldom presented.[17] In consequence, it is possible for derived terms to be introduced that have no explicit relationship to primitive terms.

Axioms

Two components of the Deduction Subsystem form the portion of the uninterpreted formal logical system used to perform deductions: A set of *axioms* and a set of *rules of inference.* An ***axiom*** is a wff to which the rules of inference may be mechanically applied, resulting in a new wff. Each axiom, by definition, has – *a priori* – some special property and that property is preserved by the rules of inference. Most often, the *a priori* property of axioms is that they are "true." Logicians often consider an axiom to be a wff that is "derived" via the rules of inference from the empty set of wffs: in other words, there is a rule of inference that, applied to no wff, permits any axiom to be introduced.

Axioms are sometimes regarded as being one of two types – logical axioms or non-logical axioms. Albeit the definitions are premature, we define a ***logical axiom*** as a formula that is *true* for all *permissible interpretations* of the formula, while a ***non-logical axiom*** is a formula that is not true for at least one permissible interpretation. We define more formally what it means for a formula to be "*true*" and the meaning of "*permissible interpretation*" in the next article.

A special property of axioms preserved by rules of inference by definition is *syntactic validity* Axioms are presumed to be *syntactically valid* wffs. A wff derived from one or more *syntactically valid* axioms via the rules of inference is itself *syntactically valid*. For the time being, we point out, and suggest that the reader keep in mind, that *syntactic validity* (discussed formally in a subsequent article) is not the same a *truth*.

It is sometimes convenient to specify ***axiom schemata*** rather than *axioms*. An ***axiom schema*** is not a wff, but instead provides the form of a wff but in which special symbols act as placeholders (a.k.a., schematic variables or meta-variables) and for which specific terms or wffs may be substituted. The symbols used as placeholders in an *axiom schema* must be defined in the sense

[17] For example, although recognizing the need, Church (1944) chooses to ignore Definitional Rules as a "complication".

that permissible substitutions are prescribed; that is, we state the domain[18] of each placeholder. Note that, if the domain of any placeholder is infinite, an *axiom schema* represents an infinite set of *axioms*.

It is desirable, though not necessary, for a set of axioms to be mutually *independent* in terms of the rules of inference*;* that is, it will not be possible to prove any given axiom from the other axioms.[19] This property is called ***axiomatic independence***. As we will see much later, axiomatic independence as applied to database theory leads to an important database design principle.

Rules of Inference

The set of ***rules of inference*** (also called rules of deduction) are a set of rules mechanically applied to a set of initial wffs (called ***premises***) and resulting in a new wff (called the ***conclusion***). We say that the conclusion is derived from the premises. Conceptually, a rule of inference acts like a function that takes one or more wffs as arguments and generates a new wff.

It is sometimes convenient to specify ***inference schemata*** rather than *axioms*. An ***inference schema*** is not a rule of inference, but instead provides the form of a rule of inference in which special symbols act as placeholders (a.k.a., schematic variables or meta-variables) and for which specific terms or wffs may be substituted. The symbols used as placeholders in an *inference schema* must be defined in the sense that permissible substitutions are prescribed; that is, we state the domain[20] of each placeholder. Note that, if the domain of any placeholder is infinite, an *inference schema* represents an infinite set of *rules of inference*.

A finite sequence of wffs, each of which is either an axiom or can be inferred from an earlier wff via a rule of inference, is called a ***proof***. In particular, the sequence is a proof of the final wff (i.e., the conclusion of the last applied rule of inference), called a ***theorem***. Each proof is generally started with at least one ***axiom*** using a special rule of inference having no premises, syntactically equivalent to an *axiom schema*.[21] A proof of a conclusion from a set of premises is a sequence of wffs terminating in the conclusion, subject to three requirements: each wff is either (1) a premise, (2) an instance of an axiom schema, or (3) the result of applying a rule of inference to earlier wffs in sequence. A wff for which there exists a proof is said to be ***provable***.

The formal proofs of proof theory require great attention to detail. By contrast, the familiar, informal proofs of mathematics and computer science are intended to convince an expert that a detailed proof is possible.

[18] Logicians often use the term “range” instead of “domain”, in the sense that the placeholder “ranges over” the members of some specified set.
[19] The relationship of axiomatic independence to the Principle of Orthogonal Design is explored in a later chapter.
[20] Logicians often use the term “range” instead of “domain”, in the sense that the placeholder “ranges over” the members of some specified set.
[21] The substitution of terms in place of the meta-variables in the axiom schema is, in effect, a rule of inference.

The Vocabulary, Formation Rules, Axioms and Rules of Inference may also be said to form the ***syntax*** of the corresponding formal logical system.

The majority of texts on logic focus on teaching, for one or more formal logical systems, the use of axioms and rules of inference to establish proofs. While familiarity with writing proofs is helpful in understanding the value and possible variety of axioms and rules of inference, such is not our objective in the present text. Rather, we seek to familiarize the reader with the components of formal logical systems, their properties, and how to select a system for some use. We will then explain how to apply that knowledge to database theory and practice.

Proofs and Proof Methods

Wffs proven within the object language of a formal logical system are called *theorems*, while wffs about a formal logical system and proven within a meta-language are called *meta-theorems*. Both theorems and meta-theorems are important. Proving that a wff is a theorem (meta-theorem) within (about) a particular formal logical system requires discovering or constructing a proof sequence, each wff of which is either an axiom of the system (meta-language) or is deducible from a prior wff in the sequence by application of one or more of the system's (meta-language's) rules of inference and the final wff of which is the theorem (meta-theorem) in question. [From here on in this section, the reader is to assume that the term "theorem" may refer to either a theorem or a meta-theorem, proven in either an object language or a meta-language, respectively.]

In many respects, theorem proving is an art. In other respects, it is a strict discipline. It is a matter of philosophical position as to whether a proof is discovered – emphasizing the lack of a specific technique that will always result in a proof – or constructed – emphasizing the disciplined form that proofs should take.

Discovering or constructing a proof of a given wff can be a rather onerous task and even elusive. On the other hand, some proofs are rather obvious and are easily found. It is often the case that more than one proof sequence may be found, some more elegant or shorter than others. Discovering or constructing proofs becomes easier with practice and as familiarity with the axioms and rules of inference of a system increases.

It is common practice to list the wffs of a proof sequence in a column along the left hand side, with annotations on the right hand side identifying the axioms, rules of inference, and prior lines relied upon. The final line of a proof, the wff to be proven, is often identified with the symbol "∴" (therefore) or "Q.E.D." (latin for "*quod erat demonstrandum*" – that which was to be demonstrated). Logic texts often provide informal proofs or arguments in paragraph form, giving an outline of the proof with the expectation that anyone competent in the rudimentary proof theory of the logical system being discussed will be able to fill in the details with a formal proof.

There are numerous methods or strategies that often lead to either proof or disproof of a wff. Familiarity with such methods will greatly enhance one's chances of finding a proof sequence leading to a particular wff. Some methods are specific to a formal logical system. Other methods are quite general and only require adaptation to the formal logical system in which they are used so that they adhere to the axioms and inference rules of that system.

Proof methods may be applied within the object language to deduce theorems, or within the metalanguage to deduce meta-theorems. A few proof methods are so important that they bear a brief discussion here so as to familiarize the reader with their form. Among these are "contingent proof", "proof by contradiction" (a.k.a., *reduction ad absurdum* – reduction to the absurd), and "proof by induction".

A ***contingent proof*** of a wff is one that assumes, without proof, that a hypothetical wff is logically valid or logically true, then proceeds to show that either the logical validity or the logical truth of the wff is derivable. It is often useful to include a contingent proof within the proof sequence of a larger proof, in which case it is called a ***hypothetical derivation***. Typically, in a formal proof, a hypothetical derivation is set off from the main proof sequence by indenting from the left and beginning each line of the hypothetical derivation with a vertical bar.

An important form of contingent proof is ***proof by contradiction***, which is also an *indirect proof*[22] as we shall see. In a proof by contradiction, the negation of the initial wff to be proved is assumed as the hypothetical wff. The hypothetical derivation is then shown to lead to a contradiction with either an axiom or previously proved theorem of the formal logical system. From this contradiction, we may consider the initial wff to have been proven. The method of proof by contradiction *per se* can only be used in formal logical systems that are consistent with the law of non-contradiction (i.e., either P or else ¬P). In systems with two truth systems, the law of the excluded middle then obtains. However, variations can be used in some (but not all) systems for which the law of the excluded middle does not hold. For example, if all wffs can be assigned only truth values that are either designated or else anti-designated, one can assumes that the hypothetical wff (negation of the wff to be proved) is designated and shows that this leads to a contradiction in terms of permissible truth value assignments (designated vs. anti-designated).

Proof by induction (a.k.a., *inductive proof*) is another kind of contingent proof, and is among the most important methods of proof. The method of proof by induction may be attempted in any situation in which a sequence of enumerable wffs exists and one wishes to show that all have some property (e.g., validity, logically true, equivalence to some wff, well-formedness of formulae, consistency, mutual independence, etc.). It therefore applies to a set of wffs. In particular, the method of ***proof by (weak) induction*** requires that one:

- prove that if an arbitrary wff ***n*** of the sequence has the property (***the inductive hypothesis***),

[22] Certain formal logical systems preclude such indirect proofs.

then the next wff ***n+1*** of the sequence has the property (the ***contingent*** or ***inductive step***)

- prove that some initial wff ***c*** (often set to ***0*** or ***1***) of the sequence has the desired property (***the basis step***)
- conclude that all wffs of the sequence after initial wff ***c*** have the desired property

The method of ***proof by (strong) induction*** requires that one:

- prove that if an arbitrary wff ***m*** of the sequence, if every wff ***k*** for ***k*** < ***m*** has the property (***the inductive hypothesis***), then wff ***m*** of the sequence has the property (the ***contingent*** or ***inductive step***)
- prove that some initial wff ***c*** (often set to ***0*** or ***1***) of the sequence has the desired property (***the basis step***)
- conclude that all wffs of the sequence after initial wff ***c*** have the desired property

Some inventiveness may be required to present the set of wffs in question as a sequence, in which the difference between wff ***n*** and wff ***n+1*** is well-defined and yet sufficiently abstract that it can be applied no matter what the value of ***n***. Here are some examples:

- We may want the set of wffs to have a common form, sequenced by some measure of complexity. For example, we might take as the set of wffs certain wffs that have a unique number of conjunction operators, the wff with one conjunction operator being wff ***1***, and the wff with ***n*** conjunction operators being wff ***n***.
- We may sequence a set of wffs by the number of arguments (for systems that include the concept) they take.
- We may sequence a set of wffs by the number of rules of inference required to derive them.
- We may sequence a set of wffs by the number of formation rules required to form them.
- We may sequence a set of wffs by some encoding that provides a unique number for each wff. *This application of the method was made famous by Gödel in his Incompleteness Theorems*.

There exist several variations on induction. The set of wffs over which the induction ranges may be either finite or infinite. Accordingly, we say the induction is ***finite induction*** or ***infinite induction***. Ordinarily, the inductive step proceeds by what is called ***inductive ascent*** – the wff sequence is conceived of as increasing with each step. However, the inductive process may also proceed by ***inductive descent*** – the wff sequence may be conceived of as decreasing with each step and progressing until the base (established by the basis step) is reached. This latter variation is sometimes easier to establish or will seem more natural than inductive ascent.

IV. ABSTRACT SEMANTICS

A formal logical system is typically developed, in the sense of selecting the alphabet, composition rules, vocabulary, formation rules, definition rules, axioms, and rules of inference, with some interpretation in mind. Such an interpretation is called the ***intended interpretation*** (a.k.a., ***canonical interpretation,*** principal[23], or standard interpretation).[24] An intended interpretation supplies an abstract semantics for a formal logical system. A formal logical system must faithfully represent its intended interpretation. Often, the intended interpretation is used as an informal explanation of the formal logical system.

For example, we usually think of simple set theory as the intended interpretation of propositional logic. Propositions correspond to set properties (which define sets) and the logical symbols for conjunction and disjunction correspond to the set operations of intersection and union, respectively.

Intended interpretations are usually purposefully abstract or non-specific. This suggests the fact that such interpretations are not completely unique (they have some flexibility) and that a large number of specific interpretations of the formal logical system are possible. Nonetheless, the intended interpretation establishes an abstract semantics with which all other interpretations must be consistent. We may say, for example, that propositional logic follows ***set semantics***. This means that *every operation and every non-logical symbol within propositional logic has a corresponding set theoretic operation and set, respectively*. Were we to import into our formal logical system something new (e.g., an operation for which no corresponding set theoretic operation existed), we would violate the intended interpretation of propositional logic. Something new (e.g., a category of symbol, operation, axiom, rule of inference, etc.) not found in propositional logic would have to be incorporated to define the new formal logical system. In consequence, the properties of the resulting formal logical system would necessarily be something other than propositional logic.

Violating the abstract semantics of a formal logical system's intended interpretation can have astounding and unexpected consequences. More often than not, those consequences are negative. For example, while the original system may have been logically consistent, all formal logical

[23] Church (1958, 1996) uses the term *principal interpretation* and speaks of *alternative principal interpretations*.
[24] To the extent that a database design can be understood as developing a formal logical system, the formal interpretation in terms of the application or subject as given by the designer is then an intended interpretation. Since this usage confuses the term with its more abstract usage, I prefer to avoid it herein.

systems having the new semantics may be logically inconsistent. We will return to this issue when we discuss relational database theory.

V. CONCLUSIONS

When we get to articles describing specific formal logical systems (such as propositional logic and first order predicate logic) and their properties, we’ll see that systems without interpretation are somewhat limited. While theorems may be proven independent of any interpretation, their validity requires an understanding of the *scope* of interpretations appropriate to the particular system. To put it another way, a formal logical system must be applied to appropriate or compatible subject matter if it is to be relevant. By compatible we mean that structures and relationships can be found in the subject that correspond to structures and relationships in the object language. In the next article we’ll see why this is the case and how one is to carry out an interpretation.