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.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

Alternative Technologies

Chapter 1: Introduction to Formal Systems
Logic for Serious Database Folks Series

by David McGoveran, Alternative Technologies

"*To any student who is not ready to... [keep in mind the distinction between object logic and observer's logic], we suggest that he ... pick some other subject instead, such as acrostics or beekeeping.*" - Stephen C. Kleene (paraphrase)

I. INTRODUCTION

In this article we will take a high level look at *formal systems*. Formal systems are intellectual tools that provide a structure for representing, capturing, and organizing knowledge, including the terms, languages and methodologies used in logic, mathematics, philosophy, science, and even the arts. A special type of formal system, called *formal logical systems*, incorporate facilities to reason formally. These endeavors may be classified as, for example, abstract or empirical, and may result in either descriptive or predictive theories. Formal systems are designed so that they can be applied to multiple subjects. When a formal system is applied to a particular subject (following a procedure we will describe later), we say it is an *interpreted formal system*. Otherwise, the formal system remains *uninterpreted* and so is abstract, a tool with the potential for many uses. We'll revisit and formalize this terminology a little later on.

The impact and importance of formal logical systems is not to be minimized. As an example of the importance of formal systems, the scientific method is meaningful only in the context of a formal logical system. The scientific method and, more generally, formalism has many detractors.[1] Much could be said regarding the positions of those detractors. I shall refrain from doing so here, taking the position that they are not directly relevant to the topic at hand. For the present, suffice it to say that I am not suggesting all intellectual progress is made in accordance with a narrow, restrictive understanding of the scientific method or that a single formal system suffices for human reasoning. Although such a formal system would be a wonderful discovery, the work of Gödel in the 1920s demonstrated that there are limits on what formal systems can do. Let me also take this opportunity to assert that the limitations on formal systems, as discovered by Gödel, Turing, Church, and others, are of crucial importance when properly applied and insofar as they are applicable at all. However, as we shall see in later articles of the series, these limitations are often poorly understood and oft misapplied.

[1] If you are interested in these discussions see, for example, the writings of Thomas Kuhn, Karl Popper, Paul Feyerabend, and even Ludwig Wittgenstein.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

Without a structure such as a formal logical system and the discipline to use it, human reasoning – regardless of the purpose or subject matter – is hardly better than either random thought or its deceptive cousin, dogma. While progress may be made toward some goal, it is likely to be slow and haphazard, with errors often repeated. Indeed, this has been borne out by the history of human progress in the absence of formal systems. By contrast, history has shown rapid acceleration of human progress when formal systems are used.[2] The scope of useful application of formal systems is enormous.

More often than not, one's first encounter with a formal system is in the context of what is commonly called a *theory*. The application of theory is generally referred to as *practice* and the person or persons applying the theory *practitioners*. Sometimes we hear the study of a formal system in and of itself referred to as *pure* or *theoretical*, while the study of the uses of a theory is referred to as *applied* or even *experimental*. As a few examples, one may study a pure mathematics or applied mathematics, theoretical physics or experimental physics, or theoretical linguistics or applied linguistics. Engineering, as a study or practice, is a term usually reserved for the application of some theory and, more often than not, consists of the application of established recipes or procedures that have been sanctioned by theory and their adaptations to specific situations. For this reason, engineering seldom takes into account all of a theory or its formal structure and reasoning, let alone its limitations. Examples include software engineering (a practice of computer science), structural engineering (a practice of mechanics), electrical engineering (a practice of electromagnetic theory), electronic engineering (a practice of electromagnetic theory and possibly semiconductor physics), and so on. As we shall see, the practice of a theory involves and depends on the interpretation of a formal system.

Practitioners in the database field sometimes demean theory (and specifically relational database theory) as being irrelevant to the practice or insist that theory should not dictate practice. Such assertions demonstrate that the relationship between theory and practice is often misunderstood, especially by those who practice software engineering. Some of this misunderstanding arises merely because the word theory is conflated with hypothesis or even a subjective understanding. Such uses of the word theory are, at best, referring to an *informal theory*. By contrast with informal theories, *formal theories* are not so easy to dismiss, involving explicit assumptions and the consequences of them.

We can assume, for purposes of simplified explanation and speaking very broadly, that there are two types of formal theories, both of which are types of formal systems. The first type of theory is called an ***empirical theory***, sometimes called an *explanatory theory*[3]. Empirical theories are

[2] Lest you think this is a comment pertaining only to technological progress, consider disciplines such as music. The development of a formal system for transcribing and creating music has its earliest known beginnings some 4000 years ago. Staff notation in use since the 16th century has led to an explosion of musical creativity, refinement, and variation, and now used the world over.

[3] Some authors differentiate between empirical and explanatory theories, but this is a subtlety that need not concern us.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

intended to be descriptive, and to have explanatory and perhaps even predictive power when applied to observations, including observations of human practices. When investigating novel empirical phenomena, one often tries to develop an empirical theory that will describe and explain that phenomena in terms of, if possible, what is already known and otherwise, to identify facts and relationships that were previously unknown. Notice that empirical theories are, by their nature, interpreted formal systems: the subject matter to which they apply is given *a priori*. To be an empirical theory, it is not sufficient for the theory to be suggested, influenced, or motivated by observation: an empirical theory must represent some specific set of observed phenomena. The familiar scientific method is a procedure intended to allow empirical theories to be developed and refined iteratively through observation, formation of hypotheses, empirical testing of hypotheses (validation and falsification), and formalization of accepted hypotheses.

Some database practitioners seem to think that database theory is or should be an empirical theory. In that understanding, they assume that the use of database technology should be logically prior to database theory. They argue that how a technology is used, including whatever works-around are necessary to circumvent flaws in that technology, should define the theory. If one were trying to develop an empirical theory of the practices of software and DBMS design and of physical database design, then it would be reasonable to understand a formal theory (and its underlying formal system) as deriving from the practice.[4] However, relational database theory was not developed as an empirical theory,[5] but a different, second kind of theory.

This second type of theory is called a ***deductive theory*** which, somewhat loosely, is a formal language augmented with rules of inference. The abstract system, devoid of any specific meaning, is an uninterpreted formal logical system, often referred to simply as a (formal) ***theory***. In practice, one interprets the symbols and terms of the language as denoting terms or elements of some subject, attempting to maximize the knowledge that can be captured in the formal language about the subject. This is a highly structured endeavor: if one violates the rules that apply to the creation of deductive theories, all bets are off and every result is forever suspect. If done in strict compliance with the rules, we can then reason (deduce) abstractly in the formal language and then translate the result into "new knowledge" about the subject. This new knowledge can then be confirmed or denied. One hopes that eventually no more "denials" occur, in which case the subject is said to be a *model* of the theory. That is, we have then provided an example subject related to the theory as specified by an interpretation, and the theory then *represents* that subject. In common parlance we often say we have "modeled the subject," but this is misleading. It erroneously suggests the theory is a model of the subject instead of making it clear that, through interpretation of the theory, the subject becomes a model of the theory.

If the latter occurs, we can often modify the theory to account for that denial. Database theories

[4] That said, I am rather skeptical of the success of such an endeavor given the unscientific way in which observations are made of the practice.

[5] Although it is true that empirical observation of then current practice influenced or motivated Codd, that is insufficient to make relational theory an empirical theory: Codd made no attempt to develop a theory of practice as it then was. Instead, he developed an abstract theory that could prescribe a new set of practices.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

(e.g., relational database theory) are usually deductive theories in the foregoing sense: they put forth a formal language and then interpret it as a theory of knowledge representation and reasoning (the general subject). The further application to some specific subject area (a domain of knowledge) requires further interpretation, a task of the data modeler, database designer, or application designer.

Our goal in this part of the series is to understand the nature of formal systems, the role of an existing formal system, the proper use of formal systems, and how to avoid certain abuses of formal systems. In other words, we want to understand the deductive theory game and its rules. As noted in the introductory article, this series will ultimately focus on applications of formal systems to database theory and practice. As we will see in this and subsequent articles in this series, there are many reasons for concluding that a DBMS and its query language should be an implementation of a formal logical system.

The relational model (an abstract "model" of data resulting in a database theory) is ostensibly built upon a foundation of four closely related formal logical systems: mathematical relation theory, simple set theory, propositional logic, and first order predicate logic. Unfortunately, many database practitioners and, sadly, many researchers, do not seem to have a solid understanding of these foundations, how they are to be combined, and how they may be used. The literature is rife with errors pertaining to the foundations of formal logical systems, especially misapplications of them to database technology and theory.

To those familiar with relational database technology, its dependence on the formal systems of propositional logic (a.k.a. Boolean logic) and set theory is probably obvious. Queries are often written and understood based on an elementary understanding of propositional logic. Less obvious, practitioners are then forced by a query language like SQL to do violence to set theory and classical propositional logic. Divergences from the underlying formal systems are embraced incidentally, with no understanding or perhaps even awareness of the consequences or their severity. For example, set theory is violated by SQL's support for bags (multi-sets) and classical propositional logic is violated by SQL's three "truth values." Having thus changed the definition of the formal system, the relied upon properties of set theory and propositional logic can no longer be assumed.

Sadly, the researchers (DBMS engineers and SQL standard committee members) that have permitted such travesties have not supplied alternatives to propositional logic and set theory in compensation. They have not specified any formal system on which SQL is allegedly based, nor have they considered the logical consequences of their decisions on the properties of such formal systems.[6] Even more alarming, well-intentioned researchers and writers on the relational data model make assertions and proposals that show a poor understanding of formal systems,

[6] One former SQL standard committee member objected to this, saying that they tried to describe what they were already given and attempt to establish some degree of standardization. I'm sympathetic. However, preserving serious errors rather than paying the cost to correct them is a misguided strategy based, at best, on short term cost-benefit analysis. At this point such errors are so compounded and entrenched that there is almost no hope of replacing SQL.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

especially formal logical systems.

In subsequent articles, we will provide a preliminary discussion of set theory inasmuch as the terminology of set theories (there are many) is useful for talking about formal systems and so will be incorporated in our meta-language.[7] We will then go on to explain what is meant by a formal logical system in some detail. In the articles that follow, we will then explain two formal logical systems – propositional logic and first order predicate logic – and discuss their properties. In addition, certain other formal logical systems and their properties will be introduced.

A goal of this series is to provide – from the perspective of formal logical systems – a foundational knowledge of set theory and logic for those interested in a deeper understanding of the relational data model, its alleged variants, databases, and database management systems inasmuch as these are (or should be) exemplars of formal logical systems.

It is not my intention in this series to create a textbook for the practice of logic (i.e., becoming proficient in the syntax and proving theorems, etc.), nor to teach anyone how to become a logician. There are many existing texts for that purpose and some are included in the bibliography. The reader, if wishing to pursue such expertise, is encouraged to study those texts. Some rudimentary knowledge of using propositional and predicate logic is assumed by this series.

It is my intention to provide enough *conceptual* understanding of the foundations of formal systems so that potential logical errors pertaining to database theory and practice become a bit more transparent … and therefore avoidable. With luck, DBMS users (especially data modelers, database administrators, and database application developers) can then understand why their commercial DBMSs fail them in so many ways. It is a further goal to provide, as a consequence of understanding formal logical systems better, useful methods for data modeling and to improve upon the elucidation and interpretation of the relational model.

Although trained in logic and its foundations, I am not a professional logician. As with previous publications, I am sure that this series will be neither as formal nor as complete as I would like to make it. I may pass over known issues. I am also certain that formal issues and developments exist of which I am unaware. Nonetheless, I have tried to assure myself (through extensive research and study) that the discussion of key issues is accurate, and that the reader will be prepared to understand the essential problems associated with reliance on a system of logic (whatever formal system it is based upon) by real-world database implementations.

We've considerable material to cover and numerous concepts to learn before applying it all to database (or other) topics. Don't be surprised if you find some of the concepts presented herein

[7] While it is true that set theories are formal systems in their own right, there is no danger of introducing circularity here as long as we differentiate the use of set theoretic terminology as meta-language and its use in the formal system under study.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

difficult to comprehend.[8] But please do try to take all the material presented here onboard. Although I've tried to motivate the necessary abstract material along the way, there is much room for improvement in this area. Be patient with me and yourself, reading all the material carefully and slowly. Give yourself every opportunity to learn. I've used various devices in an attempt to assist – including footnotes, asides, emphasis, callouts, figures, and so on. I urge the reader not to ignore these, but give them your attention. And, finally, if you find something I've written completely obscure, please let me know.

II. FORMAL SYSTEMS: SOME PRELIMINARIES

Reasoning – in all its many forms, from animal to human to computer – has been the subject of study throughout recorded history. Our understanding or lack of understanding of the reasoning process determines agreements and disagreements among individuals, races, and nations; progress and the lack thereof in the sciences from mathematics to architectural engineering; problem solving abilities; and, arguably, even our emotional well-being. Many philosophers have sought to be more careful and precise in understanding and prescribing the use of reasoning. In consequence, our body of knowledge about how to reason and what can (or cannot) be expected from it has progressively become less informal and more formal. As indicated in the introduction to this series, there are many, many reasons you need to be able to analyze and understand formal systems and their properties. While especially true with respect to database theory, it is true for most other disciplines as well.[9]

By ***informal reasoning***, I mean a thought process that is inherently non-repeatable, ambiguous, or subjective (i.e., depends on particular individuals). By ***formal reasoning***, I mean a thought process that is repeatable, unambiguous, and independent of particular individuals. By extension of these definitions, formal reasoning can be taught while informal reasoning cannot. Informal reasoning is much more likely to be flawed in its conclusions than formal reasoning. The causes of those flaws are more difficult, if not impossible, to identify. There are many methods of formal reasoning, usually reflecting a human thought process such as analysis, association, generalization and so on. The most widely studied method of formal reasoning uses inference – deriving conclusions from premises according to rules (specifically *inference rules*) – and is called ***inferential reasoning*** (or, sometimes, *deductive reasoning)*.

A ***formal system*** is a well-defined structure that facilitates formal reasoning. The present article identifies and provides general definitions of the key components required of a formal system. By "general definitions" I mean definitions that are applicable to formal systems in general, and which may be refined in various ways for some specific formal system. There are many formal systems, and of those, many that are intended for inferential reasoning. Such systems are called ***formal logical systems*** (a.k.a., *logistic systems*) and will be the primary subject of this and

[8] It is my intent that subsequent versions of these articles will add examples in an attempt to ease comprehension.
[9] I would even go so far as to assert that such understanding is relevant to the arts as well as the sciences. At the very least, it would be beneficial to know what formal systems cannot do and so what is potentially the subject of "pure" art.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

subsequent articles.

Whenever readers encounter the term “formal system”, most will probably think it refers merely to the use of a *symbolic language*. By a ***symbolic language*** is meant an artificial language constructed to express precise formulations and which uses uninterpreted symbols[10] rather than natural language words (which may have many implicit, unacknowledged connotations). Every formal system has a formal language component and formal languages are almost always symbolic languages. You will learn that a formal language is only one component of a formal system. Maybe to you, the term “formal system” suggests something hard to learn or comprehend. You might even associate the term exclusively with the use of mathematics or logic. By the time you finish reading this article, I hope that your reaction to "formal system" will have changed and you will be better informed.

The well-defined structure of a *formal system* necessarily comprises certain components (which we will introduce shortly). By a well-defined structure I mean one in which the components and their relationships are made explicit. No doubt you already have experience with some formal systems such as arithmetic, simple set theory, or elementary logic, even if you weren’t told exactly how they qualify as examples of formal systems.

While mathematics and logic are indeed formal systems, they were probably introduced to you with the intent of your learning how to apply them in fairly straightforward ways. Such introductions fail to teach the structure of formal systems and the role of each component of a formal system, leaving the student ill-prepared for any but their most rudimentary uses. Instead, the inherent informality and lack of attention to detail in such introductions often mislead, causing the student to form and internalize bad concepts and habits. Certainly those introductions offer little preparation for those needing to understand the foundations of formal systems; foundations that have been shaped by history and honed by remarkable and often subtle developments since the beginning of the twentieth century.

When we use a formal system such as arithmetic, calculus, simple sets, or propositional logic exactly as we were taught to use it, we are relying on someone else to have justified that use. Most well-educated professionals accept the foundations on which such formal systems have been built as solid, often without any knowledge of their formal properties. In fact, it may never occur to many that an alleged formal system might not be *well founded*.[11] Learning how to evaluate those foundations is not part of the typical curriculum and rarely found in texts. Yet the importance of the foundations cannot be overstated.

Suppose a structural engineer, after having used differential calculus professionally for many years, was suddenly informed that an inconsistency had been discovered in the differential

[10] Note that an uninterpreted symbol is one that has the potential for many interpretations, but has not been assigned any one in particular.
[11] You can think of a *well founded* formal system for the time being as being one that is reliable or sound.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

calculus. What would be the appropriate response? Should the engineer panic in fear that all their work was now suddenly suspect? How could they know that some obscure assumption did or did not bring their own work into question, and if it did, how serious the concern should be? Certainly the inconsistency could be a technicality or if serious, at least repairable, requiring a correcting procedure. On the other hand, it might imply that many buildings or other public structures were at risk of structural failure – possibly with great loss of human life. Surely such a possibility is disconcerting (and, hopefully, it is also unlikely).

But now suppose that, instead of a flaw in the differential calculus itself, a flaw were discovered in the foundations of the formal logical system used to reason about differential calculus. In particular, suppose that flaw were an inconsistency, meaning that you could no longer be certain what was or was not true, so that that logic's reliability were affected. Suppose further that that exact same formal logical system had been used routinely and relied upon in database theory and practice, and many, many other disciplines. How much more devastating would the potential consequences of a flaw be if the foundations of our reasoning – of our logic – were unreliable? Flaws in reasoning obviously have much greater potential negative impact than do flaws in some procedure in some domain of knowledge. Yet that is exactly what we risk when we use formal systems whose foundations we do not understand. Of course, we cannot begin to understand the foundations of any particular formal logical system until we understand, in the first place, what constitutes a formal logical system and the relevant concepts pertaining to formal logical systems.

As with structural engineering in the preceding example, competence in the selection and use of database technology requires a healthy respect for theory and its proper role. For practitioners of the database "arts", and especially those approaching it from an application developer's perspective, to scoff at theory when committing to a DBMS or a data model is an egregious error: Scoffing at database theory is rather like scoffing at aerodynamics as "mere theory" as you are about to take flight on a commercial airline. Better hope the engineer who designed the aircraft did not also scoff at theory.

Since about 1800, we have learned that the relationship between formal systems and their applications is far more complex than we had previously suspected. If care is not taken regarding which formal systems we use and how, surprising and even catastrophic results lie in wait. We might say that some formal systems are *well-behaved*, in the sense that they have wonderful properties on which users can rely even if those properties are not understood. By contrast, many formal systems are not well-behaved and there is some sense in which they cannot be relied upon. Sometimes even modest changes to the components of well-behaved formal systems result in a loss of highly desirable – even necessary – properties, and its "well-behavedness".

III. THE HISTORICAL CONTEXT OF FORMAL SYSTEMS

Our understanding of the foregoing concerns has seen rapid evolution since about 1860. Before that time, little consideration was given to the possibility that some formal system of

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

mathematics or logic might be needed, let alone that a chosen formal system might be unreliable. Although the foundation of formal systems was initially developed by Gottfried Leibniz (1646-1716), the study of the discipline was largely overlooked for over a hundred years. The consistency or inconsistency of formal systems was more or less assumed to be readily apparent. While logical contradiction, paradox, and the like were well known back to antiquity, it was often assumed the problems they represented could be contained or avoided altogether so that no damage would be done in philosophical or formal reasoning. That assumption would turn out to be naïve.

Problems encountered in using a formal system were treated simply as research problems to be eventually solved. One learned or defined a particular formal system because it was (or at least seemed to be) useful or at least interesting, deferring unanswered questions to the future and without worrying about whether or not any systematic approach to defining formal systems was in place. For example, differential calculus (invented in the late 1600s) was enormously successful as a computational device. When examined closely, the reasoning that led to differential calculus assumed several loosely defined concepts. These "holes" in the theory raised questions about its reliability. For example, continuity, so crucial to the foundations of modern differential calculus, was not provided with an acceptable and recognized definition until the 1800s by Karl Wierstrauss (1815 - 1897)[12].

The difficulties associated with properties of formal systems make those associated with the definition of continuity seem trivial. In particular, because formal systems deal with the foundations of all reasoning, how can we provide definitions of key primitive concepts without the collection of definitions being circular? Slowly but surely a philosophical debate arose. Was mathematics conceptually prior to logic or was logic conceptually prior to mathematics? Could mathematics be expressed in terms of logic or could logic be expressed in terms of mathematics? Were they possibly different expressions of one discipline? These questions could not be properly attacked until we had a foundation for formal systems.

It was not until 1879 that the German mathematician Friedrich Gottlob Frege took up the mantle of Leibniz and laid the foundations for formal systems in modern terms. Frege was the first to maintain a careful separation of logic and mathematics notationally, allowing their relationship to be examined. While the concept of an axiom had been well-known since Euclid as a formal assumption in mathematical proof, under Frege's treatment a set of axioms were now a key component of a formal method of proof – a system of deduction. (An ***axiom*** is a formula that is presumed *a priori* to be true.) A cascade of developments occurred. Charles Saunders Peirce provided (1881) an axiomatization of the arithmetic of natural numbers.[13] Frege then provided a foundation for arithmetic as a formal system in 1884.[14] Giuseppe Peano subsequently provided

[12] To be precise, both Bolzano and Cauchy provided rigorous definitions before Wierstrauss, but their work went unrecognized until after the work of Wierstrauss.
[13] Peirce, C. S. (1881). "On the Logic of Number". American Journal of Mathematics 4 (1–4). pp. 85–95.
[14] Frege, G., "Concept Notation" (1879) and "The Foundations of Arithmetic" (1884). The latter may be found in Reference 1.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

(1889) a set of axioms for the natural numbers[15] (***the Peano axioms***) based on Richard Dedekind's 1888 (albeit less precise) axiomatization[16].

Even with a foundation for formal systems, it remained a subject of speculation as to whether or not mathematics (or just arithmetic) could be expressed in terms of logic. It would be two more decades before Bertrand Russell and Alfred North Whitehead would answer the question in the affirmative, providing a logical foundation for mathematics in the seminal three-volume work *Principia Mathematica*[17], published from 1910 to 1913.

Unfortunately, other, more fundamental questions remained to be answered. In 1900 mathematician David Hilbert gave a lecture[18] in which he identified (as examples) certain unsolved problems pertaining to mathematics as being of paramount importance. He drew attention to the "similarity of logical devices" across all of what he called mathematical science and suggested that advancement would yield "sharper tools and simpler methods". With Hilbert's identification of this problem, it came to be understood as amounting to a "*crisis in the foundations"* of mathematics: If not solved, how could mathematics be trusted? Although Hilbert focused on the system of arithmetic, more general versions of several of his problems deal with the properties of a formal logical system, of great importance to the subject of this series.

Hilbert's lecture focused on the fundamental set of formal assumptions (formally called an ***axiomatic basis***) underpinning arithmetic. He called for proofs of ***axiomatic independence*** (informally, that no axiom could be derived from any combination of the others) and ***consistency*** (informally, that no axiom led by deduction to a contradiction of any other axiom). He also gave the first statement of the property of ***axiomatic completeness*** of an axiom set (i.e., that adding an independent axiom to the axiom set would not extend the system's deductive power). This notion of completeness would later be refined into a general understanding of completeness and, in particular, a concept called *deductive completeness* (defined below and discussed in a subsequent article).

Hilbert believed that a logical foundation could be provided for all of mathematics in the form of a single formal system with a provably consistent and finite set of axioms. Unfortunately, all the important schools of the philosophy of mathematics, including Hilbert's formalist school, ran headlong into logical paradoxes in tackling Hilbert's problems[19]. The search for a solution to these problems (e.g., a consistent foundation devoid of paradox) was considered so important that it became known as the *foundational crisis in mathematics*. This continuing effort arising from Hilbert's 1900 lecture resulted in a set of objectives that have come to be known as

---

[15] Peano, G. (1889). "The Principles of Arithmetic", in Reference 1.
[16] Dedekind, R. (1888). "What are and what should the numbers be?" in Reference 2, pp. 787–832.
[17] Please note that *Mathematica Principia* is Latin and that the letter "c" in Latin is pronounced like a "k" in English.
[18] I urge every would-be mathematician or logician to read this lecture: Hilbert, D. "Mathematical Problems", International Congress of Mathematicians, Paris, 1900.
[19] I will discuss logical paradoxes in some detail in a subsequent article.<TBD>

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

Hilbert's Program (1920). In summary, Hilbert's Program requires providing:

- A formal deductive language (having at least a set of axioms and rules of manipulation) capable of expressing all of mathematics.

- A *finitary* proof[20] that all true mathematical statements are provable in the formal language – a property now known as "*deductive completeness*".

- A finitary proof that all provable mathematical statements are non-contradictory – a property now known as *consistency*.

- A finitary proof that the formulae resulting from reasoning with abstract concepts become true formulae when the abstract concepts are replaced by corresponding concrete (finitary[21]) concepts – a property now known as *conservation*.

- A finitary proof that there exists an algorithm for deciding whether any mathematical statement is true or false – a property now known as *decidability*.

In case you haven't taken on board the impact of solving these problems, let's be very clear. *Were the objectives of Hilbert's Program to be met, mathematics and logic would be on such a firm footing that they could be performed mechanically with complete confidence. Only complexity would slow down the generation of results.* Remember that Hilbert's Program was introduced more than a decade before modern theories of mechanical computation, let alone their realization in computers as we understand them today. However, a solution to Hilbert's Program strongly implies that all of logic and mathematics could be relegated to automatic analysis by computer.

In 1929, Kurt Gödel[22] proved the completeness of first order predicate logic in his doctoral thesis. So far, so good. Unfortunately, Gödel's work over the next few years (1929-1931[23]) demonstrated that the objectives of Hilbert's Program, as originally formulated, were simply unachievable. Hilbert's crisis had reached a crescendo and Gödel's results were considered by most mathematicians, logicians and philosophers to be devastating.[24] So strong was the reaction that even today, the folklore and sometimes even serious literature would have us believe

[20] A finitary proof is one that, at every level of the proof, the components are finite. By contrast, certain formal systems permit proofs by a sequence of steps, each of which is finite, but for which there is no guarantee of a final step to the sequence. Another example is a proof method called *transfinite induction*, the definition of which is beyond our scope.
[21] A finitary concept is one that requires a finite number of steps to define or specify.
[22] Pronounced as in "girdle" but with the "r" essentially silent.
[23] Gödel, K. "On Formally Undecidable Propositions in Principia Mathematica and Related Systems I" (1931). in Reference 1.
[24] It is interesting that, during the same period, a similar crisis was developing in physics and the outcome was no less devastating. The crisis in physics remains unresolved today.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

erroneously that Gödel's results show that formalism is doomed. Many even *erroneously* believe that no formal system can have the properties identified by Hilbert's Program, whether capable of expressing mathematics or not. Despite shock, discouragement, and even despair, logicians and mathematicians continued work on the foundations.

We now know that all is not lost. David Hilbert and Paul Bernays published the results of their efforts on Hilbert's Program in his Foundations of Mathematics (in two volumes: 1934, 1939). Among other things, the work laid the foundation for the discipline now known as *proof theory*, an important subject to which we will return below. By modifying the objectives of Hilbert's Program slightly, considerable success can be demonstrated. For example, rather than finding a single formalism for all of mathematics, we do have a single formalism in the combination of Zermelo-Frankel set theory with first order predicate logic that suffices for what most consider to be *essential* mathematics[25]. If non-finitary proofs (e.g., those using *transfinite induction*[26]) are permitted, much more is achievable. Although Peano arithmetic (i.e., arithmetic based on the Peano axioms) is not decidable (i.e., undecidable), proofs now exist of the decidability – in the sense defined above – of both Euclidean geometry and Cartesian geometry (a.k.a. analytic geometry, or geometry using coordinate systems).

Throughout the 1900s, developments in the foundations of logic and mathematics continued. The work of Gödel, in conjunction with that of Alan Turing, Alonzo Church, Stephen Kleene[27], and Emil Post led to the development of Turing machines, the lambda calculus, and recursion theory (and ultimately to computing as we now understand it). However, these new developments raised questions as to what was and was not "mechanically" computable or "automatable". For example, we expect a DBMS to automate database tasks. But which database tasks are mechanically computable and how do we make this determination? The answer to this question should be of importance to everyone interested in database theory, especially those designing a DBMS or a language for interacting with a DBMS! The work of Turing, in particular, contained the basis for answers to such questions. In future articles, we will identify the types of formal logical systems that, in some sense, involve complexities that exceed what is mechanically computable and so must be treated with special care.

Over the same time period (i.e., the 1900s), theories regarding the concepts of interpretation of formal logical systems and logical truth were developed. Naïve notions of "truth" based on the idea that a "true" proposition is one verifiable in the "real world" (more precisely, the physical world) simply do not suffice in logic.[28] As late as 1941 when he published *Introduction to*

---

[25] Note that this combination is exactly the one chosen by E. F. Codd as a formal foundation for his relational theory. To someone familiar with formal systems and their properties, many of Codd's choices were clearly intentional.

[26] Transfinite induction is induction over the ordinals, rather than the cardinals. For more on transfinite induction, see Suppes, Patrick (1972), Axiomatic Set Theory, Dover Publications, ISBN 0-486-61630-4

[27] Pronounced "klay-nee".

[28] At best, such definitions simply push the difficulties into unanswerable questions regarding the definition of "real world" or "reality". One is led into the famous "infinite regress" of analysis, forever responding to a definition by

Logic, Alfred Tarski did not completely differentiate between deductive aspects and interpretive aspects of formal logical systems. In some respects, Tarski treated *logical truth* rather than *validity* (the idea that deduction in a formal system faithfully preserves truth under every interpretation[29]) as fundamental to the definition of logic. Although he laid the foundations in the 1930s, Tarski and his students would not fully develop the theory (called *model theory*) necessary to resolve these issues until the 1950s and 1960s. This development required the formalization of the concept of "truth". Theories of truth other than Tarski's continue to be developed and debated[30] even as of the present writing.

Defining, modifying, or critically evaluating (and ultimately accepting or rejecting) a formal system for some intended use should never be done without a solid understanding of the foundations and structure of formal systems. It is all too easy to start with a reliable formal system and end with an unreliable formal system. Nonetheless, well-intentioned researchers often do exactly that, especially in the database field where modifications to the formal system are easily established.[31] An example is the well-known transition from two-valued predicate logic to three-valued predicate logic in the database standard language SQL (properly pronounced "ess-que-ell"), a change that invalidates many of the assumed, high desirable properties of logic underlying SQL DBMSs. We will identify other specific issues in a later article.

Fragments, variants, and extensions to some familiar system of logic are proposed or used for reasoning, and mathematical (especially statistical and analytical) methods are given inappropriate interpretations (i.e., are inappropriately applied). Worse, results that pertain to the foundations of formal systems are cited in support of unrelated assertions. Among the most prevalent of all the incorrectly cited results are those of Gödel pertaining to consistency, completeness, and decidability. The literature, from academic to popular expositions, is rife with misunderstanding of these issues and their importance.[32]

The historical development of our understanding of formal systems has followed many paths and taken many turns. Individual writers have sometimes written without knowledge of others' works, introduced their own vocabularies, and attacked similar or related problems in their own way. Only with the perspective of history has there come to be some regularization to the field. Different "schools" have denigrated each other, often with vitriol. Various groups of researchers, like the Vienna Circle and the Gottingen school, pushed the field forward and had great

---

asking what the words in the definition mean. To put it more precisely, they convert questions of *epistemology* (what is known or knowable) in favor of questions of *ontology* (assertions about what exists).

[29] We'll come back to validity and interpretation in later articles.

[30] See, for example, W. O. V. Quine, "Truth by Convention" in Readings in Philosophical Analysis (editors: H. Feigl, W. Sellars), © 1949 Appleton-Century-Crofts, NY, NY.

[31] I'm not suggesting that database folks are more error prone than other experts, but rather that the field itself is closer to these issues in some sense. A similar comment could be made, for example, about knowledge representation in AI.

[32] We will try to correct these misunderstandings in a future article; however, a number of articles and much groundwork lie between this one and that.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

influence. Some writers approach their topic from the general perspective scientific theories, some from mathematics, some from logic, some from meta-mathematics, some from computing, and so on.

***Warning*:** Even when the general perspective of two sources is the same, terminology may differ and even conflict. The same terms may be used by different writers in different ways, and may even have incompatible meanings. Worse yet, writers have often tried to make their subjects "approachable" or "practical", with important concepts and subtleties glossed over or left out altogether. A partial cause of this non-standardization of terminology is that the best known texts were written while the field was still in development, and portions of it are still in development today. This varied history means that the reader of texts and scholarly articles on the subjects of formal systems, logic and its foundations, mathematics and its foundations, computing and its foundations, and meta-mathematics is likely to be confused through no fault of their own.

The present series seeks to provide and use a single and widely understood description of formal systems with set theory, propositional logic and first order predicate logic being particular exemplars. I've chosen terminology and used it in a way I consider to be more or less standard, although other writers may disagree. My goal is to help my readers understand the concepts and their relationships, and to provide a roadmap, where possible, to understanding alternative terminology through relationships rather than through distinct, possibly subjective or even imprecise definitions.

Through this strategy, it can be hoped that the reader will become critical and well-informed, and able to detect when a writer's exposition on these subjects is a gloss, suspect, uninformed, or simply incorrect. Obviously, I am not suggesting that my readers will become formalists, logicians, meta-mathematicians and the like, or be able to argue cogently with those who are. On the other hand, perhaps my readers will be able to attend such experts without being limited to reading or hearing unintelligible utterances. With luck, my readers may even be able, from time-to-time, to recognize the unintelligible or confused mutterings of non-experts when they hear them.

Many of the errors pertaining to formal systems stem from a simplistic understanding, one that ignores the essential structures of formal systems and their interrelationships. To avoid this problem, this article (and most of the articles in the series) places the structure of formal systems and their interrelationships front and center. In consequence, we will sometimes need to use terminology or introduce concepts with an informal understanding or common definition, providing a more precise definition later on. Only after the structure and interrelationships among components of formal systems have been explored will we examine specific formal systems such as propositional logic and first order predicate logic.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

IV. ON LANGUAGE

One of the concepts we need to understand early on is that of language and its importance. Obviously the use of language (verbal or nonverbal) is necessary for all communication. This series, as an attempt to communicate, is not impervious to that requirement.

Informally, a language is some vehicle for the conveyance or expression of meaning. The formal term for an individual element that conveys or expresses meaning is a ***sign***, which may be a symbol, a sound, a sight, a taste, a tactile sensation, and so on. Thus, a sign is something observed or accessed through the senses (visual, auditory, tactile, olfactory, or taste) and understood as having a meaning other than itself. A sign is literally a reference to, or "stands in place of", something other than itself. In an informal system, signs can convey or express very specific and even complex meaning.

Take note: *In the formal systems with which this series is concerned, signs are usually taken to have only abstract meaning, perhaps only sufficient to distinguish one sign from another. In a sense then, a sign in a formal system may be then understood as a potential carrier of some meaning that is to be assigned to it.*

Signs typically convey or express meaning in one of two ways: by denotation or connotation. A *sign* is said to have ***denotational meaning*** if its understanding arises from the sign "pointing to examples." Any object (including events and relationships) A is said to denote an object B if A represents (or conceptually "points to") B. B is then said to be the denotation of A, which is a *sign* by our earlier definition. For example, a name is a sign that points to the thing or concept named. A sign is said to have ***connotational meaning*** if its understanding arises from the sign "being in association with" some other sign, that association typically arising through usage. For example, we often infer the meanings of words from the contexts in which those words are most frequently used.

We will often have reason to introduce a special type of sign, called a *symbol*. A ***symbol*** is a sign whose conveyed meaning is established by convention and is typically written and visual. For example, the letters of the alphabet of a natural language are symbols, as are the marks that represent numbers. In the case of the letters of a natural language's alphabet, such symbols often denote fundamental sounds (a.k.a. *phonemes*). A ***language*** comprises an ***alphabet*** (consisting of a set of ***signs*** (usually including signs given as written *symbols*) combined to create ***terms***, a ***vocabulary*** consisting of a set of terms, and a ***syntax*** (i.e., a set of grammatical rules) for combining terms to form *expressions*. In the *formal languages* discussed herein, each sign in the set of signs will be, more specifically, a *symbol*.

The *syntax* of a language, formal or informal, may be either context free or context sensitive. A ***context free syntax*** means that expressions are built up from components according to rules that do not depend on anything other than the components to which they are being applied. By contrast, the rules of a ***context sensitive syntax*** take into account other aspects of the

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

composition. For example, if a rule can only be applied after certain other rules, or if a rule can only be applied to a component when that component is surrounded by certain other components, that rule is context sensitive. In much the same way, the semantics of a language can be either a ***context free semantics*** or ***context sensitive semantics***. If the meaning of terms or expressions in a language depends on the meaning of other expressions, or on how the expression was composed, the language has a *context sensitive semantics.* Otherwise, it has a *context free semantics*.

A language's signs may be *interpreted*, meaning that formal usage of a sign must be understood as providing an explicitly given denotation of that sign. The act of explicitly giving a denotation for a specific sign is called *interpreting* that sign, whenceforth the sign is said to be an *interpreted sign*. When all the signs of a language act as denotations, the language will be said to have a ***semantics***.[33]

All languages may be understood as being used to convey completed thoughts in either written or oral form. In most natural languages, we call these completed thoughts "*sentences*"[34]. An incomplete thought is then a partial, or incomplete, sentence. Often, the informal term sentence is used as synonymous with *utterance*, whether oral or written. In natural languages, there are kinds of sentences: declaratives or statements of belief (however strong or weak), interrogatories or questions, expletives, and so on. A more general term for a completed thought and which can be used for both informal and formal languages, is ***expression***.

Informal versus Formal Languages

By an ***informal language***, I mean a language that is perhaps in common use, but one for which the syntax is not necessarily fully defined or well-defined. For example, virtually all natural languages are informal languages. Linguists expend great effort to give natural languages a formal theory, with varied success. Informal languages, including natural languages, are presumed[35] to follow rules of composition called a ***grammar*** by which elements of the language's vocabulary (i.e., a set of words or *terms*) are combined to form those expressions. The *terms* in the vocabulary of an informal language may or may not be composed from an alphabet of signs. In some informal languages, the terms of the vocabulary are themselves the fundamental signs and cannot be decomposed.

The grammar of an informal language serves as its ***informal syntax***. In addition, an informal language often has an ***informal semantics***, meaning that it has a *partially interpreted vocabulary*. That is, at least some of the terms in the vocabulary have associated meanings (they

[33] The study of how meaning arises or is "made" is called ***semiotics***, and consists of studying the interrelationship between syntax, semantics, and ***pragmatics*** (i.e., the relation between signs and their users' acts; language behavior).
[34] *Warning*: In some formal languages, the term sentence also has a technical meaning. Which is intended is usually clear from context.
[35] Actual usage may not follow some set of composition rules strictly or the correct set of rules may not be known.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

denote something), albeit perhaps informally understood and possibly ambiguous. A language with informal semantics contains at least one or more terms whose meanings are defined imprecisely, often by loose association with other terms or context (e.g., by connotation). Most of this series will be written in an informal language, though perhaps more formal than day-to-day discourse.

Formal languages contrast with informal languages in that their vocabulary and syntax are fully defined, precise, and abstract – abstract in the sense that no particular meaning has been assigned. Typically, a formal language will have an alphabet consisting of a set of symbols. The symbols of the alphabet may be combined according to rules to form terms. The set of possible terms forms the vocabulary. An *expression* (i.e., a combination of terms) in a formal language is called a ***formula.*** A formula may or may not follow certain rules of composition of terms, called ***rules of formation***. If a formula follows the rules of formation, it is said to be a ***well-formed formula***. Most of the formal languages we will examine have a *context free syntax* and *context free semantics*.

The formal system containing a formal language under study is then sometimes called the ***object logic.*** Formal systems consist, among other components, of a formal language (used for expressions[36] within the formal system) called its ***object language***, along with a *deduction subsystem*. We might study the object logic, in which we use a distinctly identified ***meta-logic*** (sometimes called the ***observer's logic***) as the logic to reason about that object logic.[37] We then need at least one other language to reason about or describe the formal system, called a ***meta-language***. <TBD: **Figure 1.1** to be added.>

Deductive Languages, Algebras, and Calculi

Formulae in a formal language are composed from a vocabulary of terms (belonging to the language) according to predetermined rules called ***formation rules,*** yielding *well-formed formula* from *terms*. When one can form new formula from existing formulae according to predetermined rules called *inference rules*, we say the formal language is a ***deductive language*** and we call the formal system (to which that language belongs) a *formal deductive system* or ***formal logical system***. The *inference rules* are intended to preserve one or more designated properties of the expressions such as *validity*[38]and the rules may be selected or even designed for such purpose. In a sense, we can say that inference rules are a grammar used to control or elucidate some thought process using the deductive language. *Deductive languages* are an essential element of *formal logical systems*. Even if you have no experience with logic, you

[36] Reminder: An expression may belong to either an informal or a formal language. A formula is an expression in a formal language, especially a deductive language.
[37] The meta-logic may be defined in the same way as the object logic, but their uses are different and so we need to be cognizant of when one or the other is being used. We will examine this concept in more detail later and provide examples.
[38] As noted, *validity* is defined later on; for now, it suffices to know validity is a possible property of formal systems.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

probably have had experience with deductive languages.

For example, we can understand the states of play of almost any game (e.g., positions on a chess board, hands in poker, positions and scores in baseball, etc.) as the terms, and sequences of states as expressions, in a deductive language. The rules of the game govern transitions between states of play provide the grammar or inference rules. Of course, this characterization is abstract, ignoring possible elements of a game such as chance and skill. A game is a model of some formal logical system (the theory) if and only if the inference rules are sufficient to describe the transitions between valid states. Otherwise, the game is not a model for that theory, and some other theory with a different set of inference rules is relevant.

Motivations are not relevant to formal systems. For example, if one considers a game with "cheating", then additional rules are required to explain how one arrives at a state that otherwise (under the original rules) would be invalid. The addition of those rules then constitutes a formal logical system different from the one for which the original game (without cheating) was a model. For clarification, notice that there is no psychological element to the application of rules of inference. For example, whether one arrives at a valid state by "cheating" or by actually applying the appropriate rule or rules of inference is completely irrelevant. There need only exist rules of inference that could have been applied to the former state to achieve the new valid state. It is for this reason that one is usually required to state the rule of inference one is relying on when engaging in formal deduction. If one were playing a game, this would be like requiring each player to state the game rule permitting each move or play.

The point of this abstract characterization – which could be given in purely symbolic language – is that games "derive" states of play from previous states of play according to rules and any violation of a rule results in an invalid state of play. Similarly, failing to apply an inference rule properly in a deductive language can result only in an invalid formula.

Broadly speaking, deductive languages can be one of two possible types: an *algebra* or a *calculus*. The term algebra has an overloaded use in the literature, sometimes referring to a type of mathematical system having certain properties (or perhaps referring to the study of those systems) and sometimes referring to a type of language. A deductive language is an ***algebra*** if the interpretations of its formulae are restricted to *relationships* among the symbols in those formulae, and its rules are used to derive formulae interpreted as new relationships. In an algebra, the permissible interpretations of symbols is a particular class of objects, which might be a type of value (e.g., a data type) but are not necessarily a specific value. By contrast, a deductive language is a ***calculus*** if the interpretation of its formulae and the symbols used in those formulae are restricted to *specific values*, and its rules are used to compute values.

For example, a formula ($2x + y = 5y - 3z$) of elementary algebra might use symbols *x*, *y*, and *z*, each of which denote a value from the integers. Ordinarily we reason about such formula, applying various rewrite rules to derive a new formula in a desired form, such as having z and only z to the left hand side of the formula. The rewrite rules are rules of inference. At no time

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

have we evaluated the formula to determine some specific value. Even if, via the rewrite rules of elementary algebra, we were to end up with a formula of the form "variable = constant". No computation or evaluation has taken place. Technically, a formula of the form "variable = constant" can only be evaluated using a calculus or evaluation language.

By contrast, arithmetic is a calculus and involves only evaluation. Suppose that we have a formula of arithmetic such as 3*7 + 9. We can evaluate this formula according to the rules of arithmetic – for example, using the multiplication tables and the addition tables to find its value: 30. Similarly, we could evaluate 3*7 + 9 = 5*3 + 3*5 to find, by the computation, that 30 = 15 + 3*5 and so 30 = 30. We then evaluate this formula as "true" – under evaluation an arithmetic formula involving equality is either "true" or "false". As a second and third example, the formal languages of the differential calculus and the integral calculus were originally intended to produce closed formula (i.e., variables were bounded) which could be reduced to a specific value: The rules of inference of these calculi were intended for computation.

The process of "solving an algebraic equation" (a formula) involves two steps: first, deduction according to elementary algebra as an algebra *per se*, and then reducing the that algebraic formula to an implied formula of arithmetic or of logic, both of which are calculi. At that point, the arithmetic or logical formula can be evaluated, either for numeric value or truth/falsity, respectively.

A further example highlights the fact that the terms algebra and calculus are often used somewhat loosely. In developing relational database theory, Codd specified two languages: the relational algebra and the relational calculus[39]. The formula of the relational algebra express set relationships among relations (which represent data) and rewrite rules may be applied to these formula to derive a formula simplified in some sense. No specific method of evaluation is or need be given, although one is certainly implied. By contrast, the relational calculus is expressed in formulae of first order predicate logic and evaluated on specific data at every step – the formulae are said to range over certain data, for which the formulae must evaluate to "true". Although one could certainly apply the first order predicate logic rules of inference to the relational calculus (i.e., engage in deduction), it is not required – the focus is on evaluation of the truth or falsity of formulae given the data.

Confusingly, a language is sometimes called a calculus (e.g., propositional calculus, predicate calculus) in the sense that one uses the entirety to deduce ("calculate") new formula (the "values") from existing expressions if (*but only if*) the symbols in the deductive language have no assigned meaning. In the literature, this usage often occurs when authors think of a deductive language in the abstract, without considering any application *per se*. More careful authors introduce syntactic elements into the language that express abstract interpretation in purely

[39] It seems likely to me that, as is common practice, Codd simply used the terms he did because similar terms were in use: One sometimes talks of "the algebra of sets" and the "predicate calculus", these being the models for the relational algebra and the relational calculus, respectively. Hence neither is a pure algebra or a pure calculus.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

syntactic form. For example, one may add to the formal language truth-valued ***functions*** and extend the non-logical symbols with two or more symbols identified as abstract "truth values"[40]. Functions then take, for example, propositions or predicates (depending on the formal logical system) as arguments, and return an abstract truth value.

While extremely useful for reasoning about deductive systems in the abstract, this and other techniques for translating (logicians sometimes say "reducing") semantics to syntax obscures the distinctions between syntax and semantics. Such distinctions are essential when applying formal logical systems to some subject as is necessary, for example, when designing a database for some application. Semantics captured as syntax causes confusion when applying formal logical systems and we will avoid it where possible herein. When encountering a formal logical system, the reader is encouraged to determine whether or not the author has attempted to incorporate abstract semantics as syntax in the language (thereby resulting in a *calculus*). Such presentations are to be contrasted with presentation of a formal logical system as an "*algebra* of deductions," in which syntax and semantics are carefully separated and the semantics supplied as relationship between object language and subject as described herein. Separating the algebra from its semantics makes the formal process of establishing what we might call "user-supplied" interpretations easier to understand and to use correctly. Such interpretations are the essence of database design.

The reader is encouraged to differentiate between the semantics expressed by a formal logical systems' intended interpretation and any other interpretations a user may wish to establish. The intended interpretation establishes the exemplar by which the legitimacy of interpretations may be judged. Those other interpretations must not involve syntactic elements not present in the intended interpretation, nor imply relationships and operations in the subject that would be inconsistent with those permitted by the intended interpretation. For example, if the intended interpretation of a formal logical system – a call it "*ℒ1*" here – contains no dyadic operator that corresponds to equality, then a dyadic operator corresponding to equality may not be added in any other interpretation. Adding such an operator would define a formal logical system different from *ℒ1*.

V. DIFFERENCES THAT MATTER

In the study of formal systems, it is extremely important to be able to recognize when one is (or should be) using a meta-language. For example, reasoning about a formal system (especially including proving a property of the formal system or of its object language) necessarily occurs in a meta-language. The transition between object language and meta-language is rarely called out in the literature, being assumed to be obvious from context.[41] In a particular discourse, whether

[40] Church (1958, 1996) presents propositional calculus and the functional calculi of first and second order in this manner.
[41] Of course, such assumptions are likely to mislead the novice.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

an expression in a formal language is a use of an object language or of a meta-language depends entirely on how the expression is used, not on the abstract definition of the language in terms of its vocabulary and rules.

This can be confusing to the uninitiated, especially when a formal language like predicate logic serves as both the object language and the meta-language. This practice has familiar, if informal use, in English:

> *The sentence* "**This sentence is grammatically correct**" *is self-referential*.

Here, the entirety is English, with the portion shown in italics being meta-language to the object language portion shown in boldface and quotes.

The famous logician Stephen Kleene emphasized the importance of keeping object language and meta-language distinct:

> *"It will be very important as we proceed to keep in mind this distinction between the logic we are studying (the object logic) and our use of logic in studying it (the observer's logic). To any student who is not ready to do so, we suggest that he close the book now, and pick some other subject instead, such as acrostics or beekeeping."*[42]

Meta-langauges can be either informal or formal. Whether a meta-language can be informal or should be formal depends on what we want to accomplish with it. If we want to reason precisely about the object language (e.g., proving it does or does not have some property), the meta-language must be formal. A meta-language can be any language that is at least powerful enough to express all the concepts expressible in the object language. The meta-language can be even more powerful than the object language it is used to discuss. If the meta-language has more *expressive power* or less expressive power than the object language, one runs the risk of inadvertently impugning to the object language either excessive or limited capabilities, respectively.

This error is common among those whose experience is mostly confined to the use (and syntactic design) of computer languages. Assuming that a language is powerful merely because one can express or implement that language using a powerful computer language is an error of the kind Kleene warned about. For example, the language M one uses to implement a computer language L may be powerful enough to implement any computable function. However, suppose that the language L being implemented is to be limited to expressing the formula of simple set theory, which has very nice properties as a formal system. For example, programs in such a language can be debugged without human intervention to determine whether or not they preserve validity. Adding the ability to write any computable function to L makes it have the same power as M, and programs written in languages like M cannot be analyzed by any algorithm to determine

[42] Kleene, S. C., Mathematical Logic, Dover, pp. 3-4

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

whether or not they preserve validity.

For any given object language, there can be multiple languages that suffice as meta-languages. However, which meta-language is being used at any particular time for any particular purpose should always be made clear if errors are to be avoided.

With respect to this series, our general intent is to present concepts about formal systems, not to analyze them, and so the series is written in an informal language. Our general purpose and informal meta-language will be American English augmented with selected set-theoretic terminology and terms we develop or define along the way.

As we will see in more detail later, a formal logical system is often related to other systems, called ***subject systems***. Subject systems are said to be *models* of the formal logical system if certain conditions regarding interpretation are satisfied. We will discuss interpretation in detail later on. Subject systems themselves often have an associated language in which concepts and relationships are expressed, which we will call the ***subject language***.

For example, we might use propositional logic as our *formal logical system* and buying and selling apples as our *subject system*. As you may know, the operations of propositional logic correspond to operations on sets and the propositions of propositional logic correspond to sets. Our subject system includes operations of buying apples and adding them to our apple inventory or removing apples from our apple inventory and selling them. We can relate the two systems by establishing relationships between them. In particular, we can use "B", "S", and "I" of the formal logical system to represent sets, interpreted as corresponding to sets of apples bought, sold, and in inventory, respectively, of the subject system. The operation in the subject system of buying apples and placing them in inventory corresponds to the operation of "B UNION I" in the formal logical system. The operation in the subject system of removing apples from inventory and selling them corresponds to "I MINUS S".

To expand upon Kleene's edict, uses of each of the meta-language, object language, and subject language must be kept distinct. The confusion that results from creating expressions that indiscriminately mix terms from these three languages cannot be overstated. Such confusion has been studied by and caused grief to many great logicians and philosophers. Frege, Russell, Quine, and others have expended enormous efforts on the subject.

The reader would be well-advised to consider a few conclusions, listed below, regarding usage problems pertaining to terms. Every term or symbol is a sign that stands for or refers to something, however abstract or concrete, called its ***referent***. When using terms one must be careful to distinguish:

- *use* versus *mention* – Ordinarily, the appearance of a term in an expression constitutes "usage". By contrast, the appearance of a term in an expression in which the term itself is the subject matter is called a "mention". A term that is a mention is properly identified in

quotes: “term”. For example, in the sentence below the first appearance of the term “Richard Nixon” is a use while the second appearance is a mention.[43]

“Richard Nixon signed ‘Richard Nixon’ on important documents.”

- *definiendum* versus *definiens* – A term (the definiendum) and its definition (the definiens) form a relationship. In general, a definition does not supply an equivalence relationship in the sense that the definiens can be substituted for the definiendum. They are not the same thing. Sometimes the definiens is sufficient to identify the definiendum uniquely. In common usage, however, the definiens is rarely so precise. For example, although a specific set may be defined by the properties its members possess, the set of properties and the set are not identical (as the term set is normally understood in set theory).

- *type* versus *token* – A term may represent a type (class, set, or concept) or it may represent a token (or instance, member, or example) of a type. A type is never the same as its exemplars. For example, a set with exactly one member is a distinct concept from the one member.

- *name* versus *referent* – A symbol is distinct from the concept to which it refers or *denotes*. This problem is similar to a named variable in computing referencing a value or an address (pointer). Part of the difficulty with translation is that natural languages provide (somewhat ambiguous) support for referent nesting – a referent can be a sign for another referent which can be a sign for yet another referent and so on. This is similar to the problem of pointer resolution in computing, but with the added problem that each level of indirection may yield an entirely different semantic context.

- *non-equivalence of signs* – Multiple distinct signs that identify the same referent are often found in human discourse. It is a common logical error to assume that these signs are identical just because they identify the same referent. The usual, but difficult to satisfy, test of equivalence of signs is a requirement that the *evaluation* of every expression remain unchanged under substitution of one sign for the other. This same concern may be applied to anything that has a referent including symbols, terms, and formulae. For example, from “Richard Nixon signed ‘Richard Nixon’ to many important documents.” and “’Tricky Dick’ refers to the same person as ‘Richard Nixon’.” it does not follow that “Richard Nixon signed ‘Tricky Dick’ to many important documents.”

- *signs with multiple referents* – Sometimes the same sign can be used to identify different referents, with context serving to distinguish quite different *mentions*. For example, the name “John Kennedy” may be used to identify a former president of the United States.

[43] For a more detailed use of this example, see Hill, Claire Ortiz. Rethinking Identity and Metaphysics: On the Foundations of Analytic Philosophy. © 1997, Yale University Press, New Haven and London.

However, if we say that someone signed a document "John Kennedy", the referent is a signature, which itself might be understood as a sign for the previously mentioned individual. This same concern may be applied to anything that has a referent including symbols, terms, and formulae.

These distinctions are extremely troublesome. It helps to use different notations or fonts for the various languages we've introduced (meta, object, subject), and within those languages to use some distinctive annotation to distinguish between a type and a token (e.g., upper case versus lower case, bold versus italics, etc.). If multiple models are in play, each should be given a distinct notation to avoid confusion.

One final caution is appropriate regarding terminology. The following often have a confused explication (if explained at all) in the literature:

- name equivalence – equivalence among symbols, terms, or variables, which is a logical property
- referent equivalence – equivalence among values, which is a semantic property
- instantiation – selection of a particular token (a.k.a., instance) of a type
- symbol definition – an interpretation of a symbol
- value assignment – an evaluation following a prescribed evaluation procedure

We will return to these concepts and their differentiation at various points later in the series. For now, the reader is advised to put aside (or at least reconsider) definitions they may have read elsewhere regarding the foregoing terminology. The definitions provided herein will have priority.

VI. RELATING FORMAL LANGUAGE AND NATURAL LANGUAGE

Rendering an expression in one language into an expression in another language is, of course, called translation. If the two languages are of equal expressive power, then a one-to-one translation can – at least in principle – be effected. However, in the literature on formal logical systems, *translation* usually implies that one of the languages is formal and the other informal, usually a natural language. Very often, for example, we may need to translate between a formula of a formal logical system's object language and either an *informal meta-language* or else an *informal subject language* for a particular model. Similarly, students learning a formal language are often required to practice the process of translating a natural language expression into a symbolic expression or "reading" a symbolic expression as if it were a natural language expression.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

Translation between formal and informal languages, or between languages of unequal expressive power, is necessarily imprecise. The process of translation (whether to or from a formal language) is aggravated by several characteristics of natural languages, including for example:

- Both the syntax and semantics of natural languages are usually context dependent.
- Context can be conveyed through one or more various mediums – culture, situation, intent, emotional state, non-verbal signals, conversational history, expressive refinement, participant histories, and so on.
- The denotations of natural language expressions or terms are often ambiguous.
- Both the syntax and semantics of natural languages can be order sensitive.
- The referent of a pronoun may be difficult to resolve.
- Connectives in natural language do not behave like the operators found in formal languages. For example, the conjunction "and" in English may mean "and" in the normal conjunctive sense, "and then" in the temporal sense, or "and in consequence" in the causal sense. Similar concerns apply to "or", "implies" and "not". [44]
- Truth and falsity are easily confused with, for example, belief and disbelief, assertion and denial. This mistake is often made by database folks, who think a database reflects the state of the physical world, rather than representing belief or assertion about it.
- Natural languages rarely distinguish use versus mention, type versus token, or object language and meta-language.
- The formation rules of natural language grammars are not exhaustively known, nor are they stable.

Which of these are relevant depends on the natural language, the expressiveness of its vocabulary, and its grammar. Similarly, the relative expressive power of the formal language and the natural language will determine whether or not concepts and relationship in one can be expressed in the other, assuming the natural language expression can be precisely understood.

[44] Chomsky tried to resolve these differences (and others) by postulating that natural languages have a deep grammar (wherein these ambiguities are resolved) and surface grammar (where these ambiguities appear). This theory fails, in part because no cues exist in the surface grammar to guide the deep grammar resolution and in part because there exists no formal system in which the observed surface grammar can be derived from the proposed deep grammar.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

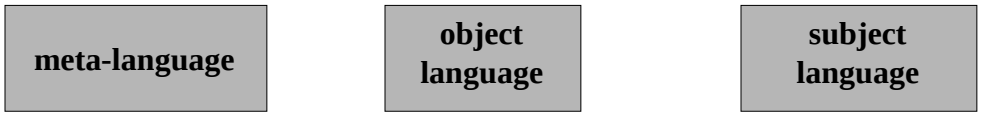

Figure 1.2: Translation

Translation will often be required between the meta-language, object language, and subject language (see Figure 2.2) in the course of studying formal systems. Sometimes, an informal rendering will be an initial result. The reader should understand that the results of an informal rendering do not belong to the formal logical system's object language, any more than a formal rendering of a natural language expression belongs to the natural language. Translation is an inherently imprecise process, primarily because it involves asserting a relationship between an informal language and a formal language. One is always in the position of *asserting* rather than *knowing* a degree of equivalence, syntactic, semantic, or (hopefully) both. An expression and its translation must never be treated as identical[45].

As an example of the uses of translation, consider the study of syllogism. Historically, a ***syllogism*** is understood as a three statement argument – a major premise, a minor premise, and a conclusion – expressed in a natural language. One way of studying syllogisms and philosophical arguments is to translate them into a formal logical system. Such efforts are fraught with the perils of translation and the source of endless debate in the literature.[46]

VII. THE STRUCTURAL COMPONENTS OF FORMAL SYSTEMS

As you begin to read this series, keep the following facts about the structural components of formal systems in mind, introduced without explanation here. We'll get to definitions and more explanation in later articles.

- The study of formal systems involves two distinct languages: an object language and a meta-language.
    - An *object language* is used for expressions within the formal system.
    - A *meta-language* is used for expressions about the formal system.

[45] By "identical" here, I mean that the expression cannot be replaced by the translation in the object language, nor can the translation be replaced by the subject language.

[46] The author's first formal college course in logic led to an attempt to translate Plato's Phaedo into first order predicate logic, which I chose as the deliverable for an advanced self-study course under my logic professor's tutelage. It exposed fallacies and rhetoric, while teaching me how difficult the process could be. The work was misplaced decades ago.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621

- A formal system can be either uninterpreted (i.e., stands alone) or interpreted (i.e., is related to another system, called the *subject*).

- A formal system may be of several types, including (1) a formal system of representation (e.g., a descriptive theory) and (2) a formal system of deduction (a.k.a., deductive or inference system) such as a formal logical system (a.k.a., a logistical system).

- In principle, a formal logical system has a *deduction subsystem* and *an interpretation subsystem*, each completely separate from the other except as intentionally related. The interpretation subsystem – though conceptually a part of every formal logical system – comes into play only when the formal logical system is to be interpreted.

    - The ***deduction subsystem*** is used for *expression* and *deduction* (a.k.a. inference), thereby formalizing the *syntax*. The language of expression and deduction of a formal system is called its *object language*.

        - ***Expression*** is the formal process of creating syntactically correct formulas in its formal language (the object language). Expression is necessary for a *formal system of representation*.

        - ***Deduction*** is the formal process of deriving a formula from other formulas. It is a necessary part of a *formal system of deduction*, but not of a *formal system of representation*.

        - The study of deduction subsystems is called ***proof theory***.

    - The ***interpretation subsystem*** is used for *interpretation* and *evaluation*, thereby formalizing the *semantics*. The language of interpretation is a *meta-language* and the language of evaluation of a formal system is called its *evaluation language*.

        - ***Interpretation*** is the formal process of relating – through *meaning assignments* - the formal system to another system called a *model*. A model is an application of the formal system. <TBD: example.>

        - ***Evaluation*** is the process of calculating the value (e.g., truth or falsity in a formal logical system) of interpreted expressions of the formal system through value assignments (*truth assignments* in a formal logical system), followed by a procedure yielding a value for the overall expression. It is performed meaningful only for an interpretation (or class of interpretations). Evaluation is done in an *evaluation language*.

        - ***Meaning assignments*** (according to rules of correspondence) are

fundamentally different from *truth assignments*, even though both pertain to interpretation of a formal system.

- A ***subject language*** is used for expressions *within* a model – that is, within the subject system that serves as an interpretation of the formal system.

- The study of interpretation subsystems is called ***model theory***.

VIII. CONCLUSIONS

Hopefully, you will be comfortable with each of the foregoing facts by the time you finish reading the portion of this series devoted to formal logical systems and before you begin the portion devoted to database theory and practice. As you read, keep coming back to these statements. Ask not only what they mean, but why they should be true. When you have these concepts well in hand, you'll be equipped to understand the relationships between the components of a formal system and thereby the properties that characterize a formal system, distinguishing one formal system from another formal system.

In the next few articles, we'll examine the structure and components of formal logical systems in more detail. First we'll learn the terms we need from set theory to be able to talk about formal logical systems. Then we'll delve into the components of the deduction system – what most texts treat as being the entirety of a formal logical system (although they also incorporate a few elements of the interpretation system without acknowledging them as such). Finally, we'll examine the components of the interpretation system.

www.AlternativeTech.com mcgoveran@AlternativeTech.com

Post Office Box 4699, Deerfield Beach, FL 33442 Telephone: 831/338-4621