NATURAL LANGUAGE QUERYING OF HISTORICAL DATABASES

James Clifford
Department of Information Systems
Leonard N. Stern School of Business
New York University
44 West 4th Street, Suite 9-170
New York, NY 10012-1126
(212) 998-0800

May 1987

Working Paper Series
STERN #IS-87-37

Abstract

In this paper we examine the connection between two areas of semantics, namely the semantics of historical databases and the semantics of natural language querying, and link them together via a common view of the semantics of time. Since the target application domain is an historical database, we present the essential features of the Historical Relational Database Model (HRDM), an extension to the relational model motivated by the desire to incorporate more "real world" semantics into a database at the conceptual level. We then present the essential features of QE-III, a formally defined English database query language whose semantic and pragmatic theory, based on a Montague-type semantics, makes explicit reference to the notion of denotation with respect to a moment of time. We demonstrate the use of this language to query an example historical database, and discuss the issues of how to provide both a semantic and a pragmatic interpretation for questions within a model-theoretic framework.

1. Introduction

The relational model of data (RM), first proposed in 1970 [Codd 70], has by now become the standard for both database practitioners and theoreticians alike. In spite of this success, however, much recent database research has focused on ways to **extend** RM to overcome perceived shortcomings. Chief among the criticisms has been RM's lack of any "real-world semantics." Among the many, diverse efforts directed at this deficiency have been a number of attempts to extend RM to incorporate a temporal dimension at the model level. While such efforts as [Ben-Zvi 82], [Ariav et al. 84], [Snodgrass 84], [Lum et al. 84], [Clifford 85], [SnodgrassAhn 85], [GadiaVaishnav 85] have all addressed this issue, the Historical Relational Database Model (HRDM) ([Clifford 82a], [CliffordWarren 83], [Clifford 85], [CliffordCroker 87]) has the advantage of being directly parallel to a formal theory of natural language. In Section 2 we present an overview of HRDM, as it serves as the environment in which we wish to explore our query language. In particular, HRDM views database attributes as **functions** from moments in time to values (in the appropriate domain), and the intensional logic IL_s provides a mechanism for direct reference to these "higher-order" objects, and for incorporating them into a general temporal semantics for the database. We can therefore express both static and dynamic queries in the same language, by quantifying over variables of the appropriate types.

In a series of papers culminating in [Montague 73], henceforth PTQ, Richard Montague embarked upon a program of providing a formal syntax coupled with a model-theoretic semantics for increasingly sophisticated fragments of English. Section 3 argues that a successful formal treatment can be given to a Natural Language querying facility for an historical relational database (HRDB), through the medium of

the intensional logic IL_s.

We view this work as important for two very different reasons. First it represents one of the first attempts to adapt the ideas of Montague Semantics (MS) ([Montague 74]) to a practical problem ([Landsbergen 81] looks at the issue of Machine Translation within an MS framework.) The research that has been done since the PTQ paper has primarily looked at extensions or modifications to its linguistic or logical theory, or at implementations of the theory on the computer. We will attempt to show that this theory of language can serve as the formal foundation of a useable computer system for querying actual databases.

Second, in addition to approaching the problem of NLQ formally, rather than from a purely engineering approach, the theory presented provides a novel (but see [Gunji 81] for a similar approach developed concurrently with ours) approach to the interpretation of queries that involves both a semantic and a pragmatic account. This work represents only a first step in this direction within a MS framework. The fragment of English which we define herein is certainly not adequate to express all of the queries that one would want to present to an HRDB. It is intended only to lay the groundwork for a formal theory of database querying that is both extendible and implementable.

In this paper we present an informal overview of a fragment of English for database querying which we call QE-III. We discuss the kinds of properties and abilities that a database query language in English should possess; principal among these are (1) an account of question semantics that possesses close analogs in database theory, (2) an account of the semantics of multiple-WH questions, (3) an account of the semantics of time, and (4) a grammar that is conducive to a computer implementation. After examining a number of partial solutions to these problems, we introduce the notion of a formalized pragmatics as an equal partner with the syntax and semantics in the specification of the QE-III language We argue that assigning to the pragmatic component the task of providing a representation for the answer(s) to a question is both appropriate and elegant. Finally we discuss several other recent attempts at developing a formal theory of questions.

QE-III is defined as a **formal** language, with syntax paired with semantics, and with a pragmatics defined on the two of these; the language as a whole is designed with the database application in mind. QE-III is both a simplification and an extension of the PTQ semantic theory. Within the tradition of

Montague Semantics, QE-III is a formalized fragment of English allowing *questions*, *tenses*, and *temporal operators*. The inclusion of a formal pragmatics as an interpretive component of QE-III is an interesting extension to the traditional conception of a Montague Grammar. Among the other extensions to the PTQ fragment embodied in QE-III are (1) the inclusion of time-denoting expressions and temporal operators, (2) an analysis of verb meanings into primitive meaning units derived from the database schema, and of course (3) the inclusion of certain forms of direct questions. These extensions, and the semantic and pragmatic interpretations with which they are provided, are motivated by the ultimate goal of database access, but they are equally interesting in their own right. The syntactic theory presented is in some cases admittedly naive, for we have been primarily interested in getting the interpretation right.

Section 4 provides an overview of the salient features of the QE-III by means of a number of example derivations and translations. The complete definition of QE-III is given in [Clifford 82b] and again in [Clifford 87] where it appears with a fuller set of examples. We conclude in Section 5 with a discussion of some of the limitations of the fragment and of some possibilities for further extensions.

2. The Historical Relational Database Model

Analogous to the relationship between the relational model of data and first-order logic ([GallaireMinker 78]), we can view an HRDB as a model for $\mathbb{L}_s$. ([Clifford 82b]). The higher-order language $\mathbb{L}_s$ (with its built-in concept of denotation with respect to an index) provides a formal semantics for such databases in a natural way.

In the standard or "static" relational model, we might see a relation such as *emp* on a scheme EMP(EMP-NAME MGR SAL DEPT). A typical query of such a relation, say "What is employee John's salary?", would be expressed in the relational algebra as $\pi_{SAL}\ (\sigma_{EMP\text{-}NAME=John}\ (emp))$. A first-order language would express this same query as something like $\{z \mid \exists x \exists y\ emp(John,x,y,z)\}$ where x,y, and z are individual variables and John is an individual constant. To answer such a query, a Data Manipulation Language (DML) would access the current relation instance *emp* on EMP, such as the one in Figure 2-1.

More complex queries about the employees in this company, such as: (1) "Has John's salary risen?", (2) "When was Peter re-hired?", (3) "Did Rachel work for the toy department last year?", (4) "Has John ever earned the same as Peter?", or (5) "Will the average salary in the linen department surpass 30K within the next 5 years, if current trends continue?" have typically **not** been expressible in any query

EMP	EMP-NAME	MGR	DEPT	SAL
	John	John	Linen	25K
	Mike	John	Linen	17K
	Elsie	Elsie	Toy	26K
	Liz	Liz	Hardware	30K
	Rachel	Liz	Hardware	29K
	Peter	Liz	Hardware	29K

relation emp

Figure 2-1: Example Relation

language, because neither the **structures** nor the **operations** in the underlying data model provide for them.

In practice, database administrators have had to resort to *ad hoc* solutions, typically involving programming in some host language, in order to handle queries of this sort. The issue of modelling time in a DBMS has recently attracted considerable attention within the database community. The HRDM (and other historical or temporal database models) attempts to satisfy the need for access to the temporal dimension of information by providing a unified and formal theory of database semantics that includes time. In particular, HRDM and QE-III, recognizing the need for maintaining an historical record of changing data, and a language (English) that makes (explicit or implicit) reference to the concept of time, together provide a theory of database semantics capable of interpreting sentences in the language correctly, i.e., in a way that corresponds with our intuitive understanding of the relation of time to the semantics of the real-world.

Consider again the query "Has John's salary risen?" Even with time represented explicitly in the database, there is no apparent simple relational algebraic formulation for this query. With the first-order representation for John's salary given above, as a first guess we might imagine that RISE($\{z \mid \exists x \exists y\ emp(John,x,y,z)\}$) would represent this new query, where RISE is a predicate symbol. However even with the knowledge that John has only one salary, say 25K, it clearly makes no sense to ask whether 25K "rises." To answer this question, more data is needed than the current **extension** of John's salary: the values of John's salary for some other point(s) of time (in this specific instance, in the past) are needed. The HRDM model presented in [CliffordCroker 87], built upon a formalization of the concept of **intension**, provides a uniform way to view attributes (such as SALaries) not as individual dollar amounts, but as *functions* from moments in time to dollar amounts. For the purposes of this paper we will present an overview of HRDM and discuss some of the issues involved.

Informally speaking, tuples in a relation represent facts about some "object" (entity or relationship) identified by the value of the **key** attribute(s). For example, in relation emp on scheme EMP (EMP-NAME MGR DEPT SALARY), the attribute EMP-NAME is the key attribute, and DEPT, MGR and SALARY define properties of employees. A particular tuple, e.g. <Peter, Hardware, Maria 30K> represents facts about the employee Peter. A relation in the ordinary, or "static" relational data model, would consist of a set of such tuples representing the facts about a set of employees. Each tuple would consist of exactly three atomic values, one for each of the three attributes in the scheme.

By contrast, in HRDM a relation would provide historical information about the changing values of the attributes of the **objects** denoted by values of the key, in this instance about EMPloyees. Each tuple would be a complex, three-dimensional object whose "size" would be based upon what we call the **lifespan** of that particular employee, i.e., the times when that employee was of interest to the enterprise. Figure 2-2 depicts two tuples in the same relation but with different lifespans.

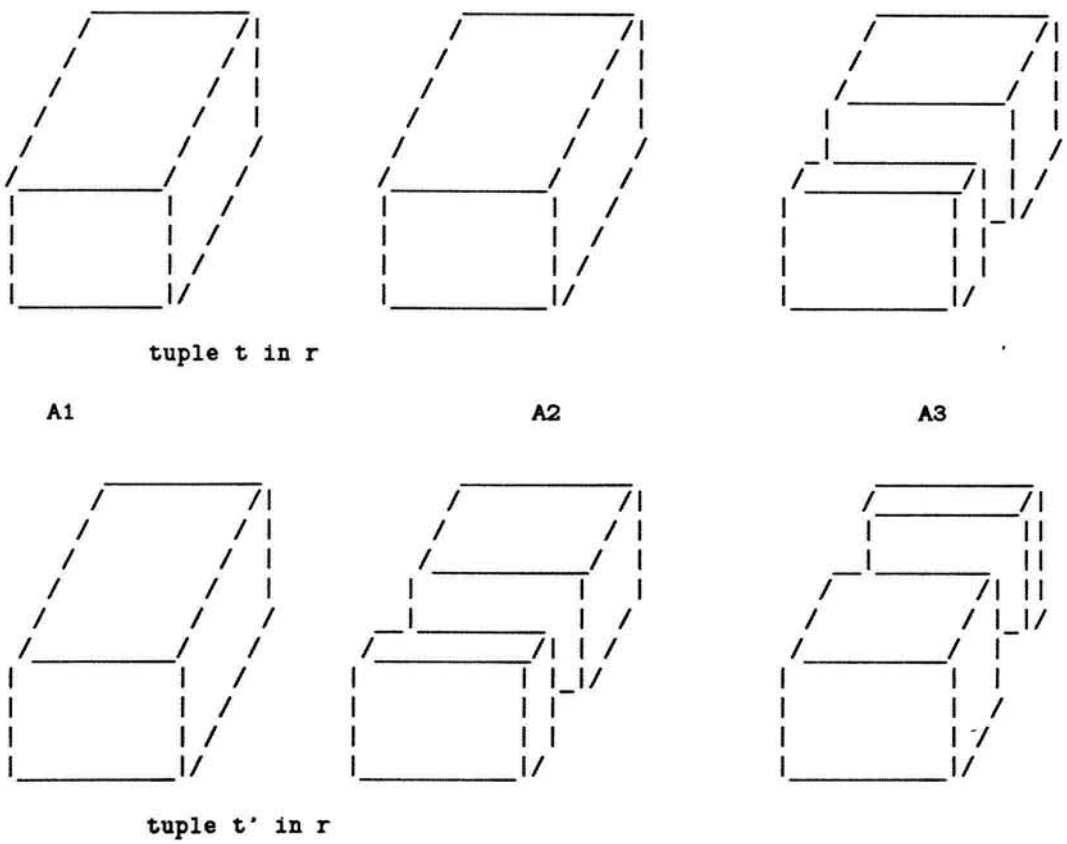


Figure 2-2: The history of two similar "objects" represented as two tuples in an historical relation

Time is represented in the HRDM as a set $T = \{ ..., t_0, t_1, ... \}$, at most countably infinite, over which is defined the linear (total) order $<_T$, where $t_i <_T t_j$ means t_i occurs before (is earlier than) t_j. (For the sake of clarity we will assume that $t_i <_T t_j$ if and only if $i < j$.) The set T is used as the basis for incorporating the temporal dimension into the model. We assume that T is isomorphic to the natural numbers, and therefore the issue of whether to represent time as intervals or as points is simply a matter of convenience. Using the natural numbers allows us to restrict our attention to closed intervals (a closed interval of T, written $[t_1, t_2]$ is simply the set $\{t_i \mid t_1 <= t_i <= t_2 \}$).

$D = \{ D_1, D_2, ..., D_{n_d} \}$ is the set of **value domains** where for each i, $D_i \neq \phi$. Each value domain D_i is analogous to the traditional database notion of a domain in that it is a set of atomic (non-decomposable) values. In HRDM, however, attributes take their values not from these simple domains, but rather from more complex functions. $U = \{ A_1, A_2, ..., A_{n_a} \}$ is a (universal) set of **attributes**. Simplifying somewhat, we define over the sets T and D a set of temporal mappings from the set T into the set D. This set, $TD = \{ TD_1, TD_2, ..., TD_{n_d} \}$ where for each i, $TD_i = \{ f_i \mid f_i : T -> D_i \}$, is the set of all partial functions from T into the value domain D_i.

The domain of each attributes in HRDM is some set of partial temporal functions. Since key attributes are intended to be time-invariant, they are constrained to take a constant-valued function (i.e., one which associates the *same* value with every time in its domain) as their value. As we shall see, these mappings are the counterparts to the notion of **individual concepts** in the intensional logic IL_s.

The notion of a **tuple t on scheme R** is expanded in HRDM to be an ordered pair, $t = <v,l>$, where

1. t.l, the **lifespan of tuple t**, is a subset of the set T, and represents the set of times over which its attributes are defined, and

2. t.v, the **value of the tuple** is a mapping such that $\forall$ attributes $A \in R$, t.v(A) is a mapping in t.l –> DOM(A) (the value-domain of attribute A).

2.1. Example Database

In the remainder of the paper we will discuss the semantics and pragmatic theory of QE-III, illustrated with example database queries to an HRDB. For this purpose we now define the relation schemas for an historical department-store database based upon an example in [Chang 78]:

EMP_REL (EMP MGR DEPT SAL)

DEPT_REL (DEPT FLOOR)

ITEM_REL (ITEM TYPE)

SALES_REL (DEPT ITEM VOL)

This concludes our brief overview of HRDM. For further details the reader is referred to [Clifford 85] and [CliffordCroker 87].

3. Overview of English Query Language QE-III

3.1. Introduction

HRDM serves to formally incorporate a temporal semantics into an extended relational database model. In order to query an historical database using English, we define the semantics of queries expressed in English in terms of the semantics of HRDM, by defining a small query fragment as a Montague Grammar. The correlation between the database semantics and this query language is made explicit by providing the semantics of the query fragment via an indirect translation into the intensional logic $\mathbb{L}_s$.[1] The translations provide for a completely extensional treatment of verbs, (i.e., there are be no verbs like "seek" which can be nonextensional in object position in the PTQ treatment). This treatment is dictated by the application to a database environment, in which existence is tantamount to existence in the database ([Reiter 78]). Through these translations, then, the historical database essentially serves as a model for $\mathbb{L}_s$ and therefore as the model for a formal definition of the interpretation of the English queries. In addition to providing a semantic interpretation, which in model-theoretic terms is called its **denotation**, we also provide for each expression a **pragmatic interpretation** in a manner to be explained.

Our goal in this effort has not been to define an English database query language that is, in any sense of the term, complete. Rather we have been motivated by two complementary goals. First, we have wanted to investigate the possibility of a formal, model-theoretic query language for historical databases. This led (somewhat) naturally to our interest in Montague Semantics and to our second goal, demonstrating that

[1] See [Clifford 87] for the definition of $\mathbb{L}_s$, and a discussion of how (and why) it differs from Montague's $\mathbb{L}$.

Montague's theories of natural language semantics are applicable to such a practical task. Along the way we discovered that it was simpler and more natural to define the interpretation of this query language in two components, one semantic and the other pragmatic.

Two overriding principles have guided this work. First was that whatever interpretation or "meaning" our theory would give to a natural language database query should be as close as possible to the interpretation given to database queries in, say, the relational algebra or calculus. This meant that the interpretation of a query should somehow encompass its *answer* as represented in the underlying database. Second, the theory should make sense computationally. This meant taking into account what had already been learned about parsing strategies for Montague Grammars [FriedmanWarren 78], [Warren 79], [Landsbergen 81], as well as what database theory had to say about the semantics of the modelled enterprise. These principles motivate certain systematic simplifications to the PTQ translations from English to logic , wherever these are suggested by the simplified view of the semantics of the enterprise provided by the database model. Moreover, since we are not attempting to develop a semantic theory of questions for English in general, these simplifications have been introduced into the translation process as early as possible. We believe that this strategy has the dual effect of making some of the PTQ theory a little more accessible, and eliminating the need to resort to the less computationally attractive technique of introducing a large number of Meaning Postulates and using logical equivalences to perform the reductions at a later stage.

We have made little attempt to develop a sophisticated syntax for our fragment. Numerous extensions to the syntax of the PTQ fragment have been investigated by researchers in the past decade that we have not incorporated into our fragment. Since our primary concern has been "getting the meaning right," we felt that a too broad **syntactic** coverage might obscure our major points. For this reason we have extended the PTQ fragment only slightly. The treatment of questions that we present is syntactically naive, although in its favor we might point out that, unlike most work on questions in Montague Grammar QE-III, makes a stab at direct questions. We believe that the semantic theory of questions that we present, and particularly our proposal to capture the answer in a pragmatic component, are an important contribution to the formalization of the interpretive component of natural language understanding systems. Naturally the true test of a "natural language" query facility is in how useable it is; certainly the syntax of QE-III would have to be extended before anyone would think of using it.

In this section we discuss the major issues underlying the definition of QE-III, which fall roughly into two broad categories: aspects of the process of database querying that we have incorporated into the fragment, and modifications and additions to the PTQ fragment that these, and the database semantics, have occasioned. As in much of the work that has been done in the area of Montague Semantics since Montague's death in 1970, we have allowed the PTQ fragment to stand pretty much intact as the heart of QE-III. However we have re-defined this fragment in terms of the language IL_s, in order to allow **direct** reference to moments in time.

3.2. Preliminaries

3.2.1. Individual Concepts vs. Entities

Most recent research in the field of Montague Semantics has incorporated the suggestion, first made in [Bennett 74], that Montague's treatment of common nouns (CNs) and intransitive verbs (IVs) as denoting sets of individual concepts (ICs) is unduly complicated. Under Bennett's suggestion both CNs and IVs denote sets of simple individuals, with the result that the entire typing scheme of the English categories in these fragments is considerably simplified. In Section 2 we showed that attributes in an HRDB can be identified with ICs. Accordingly we have not adopted the Bennett type system, but have instead maintained the treatment of PTQ.

3.2.2. Verbs

Montague's semantic treatment of verbs leaves them completely unanalyzed; thus, for example, the English verb "walk" translates into the constant "walk'" in IL, "love" into "love'", etc. The interpretation of these constants is some function in the model for the language, a function about which Montague says nothing except to specify its logical type (and in certain cases to specify an extensional Meaning Postulate). Because we are using a database as a representation of the logical model we are in a position to provide an analysis of English verbs that takes into account the meaning of the verbs as encoded in the database. This analysis is given in terms of the database schema. For example, instead of translating the verb "manage" into the unanalyzed predicate "manage'", we take advantage of the database semantics to incorporate directly into its translation the information that its subject must be an IC in the role of a MGR, and that its object must be a constant IC that is an EMP. We do not change the logical type of the translation, i.e. a transitive verb in our fragment denotes the same kind of function as it does in Montague's treatment; we simply analyze its meaning in terms of the database primitives.

This analysis in terms of a small set of primitive meaning units is not very different from some approaches taken in AI work in natural language understanding (e.g. [Schank 72]), or from the linguistic theory of deep cases [Fillmore 68]. The difference, of course, is that our primitives or cases are different, motivated by the HRDM and the schema design, and are no more absolute than any well-chosen database design.

As an example, the translation of "manage" in our fragment is given as:$\lambda W \lambda x W(i)(\lambda y AS\text{-}1(y(i),x) \wedge EMP_{\bullet}'(i)(y(i)) \wedge MGR'(i)(x))$. This expression is of the same logical type as manage' in a PTQ-like treatment, and will combine with Terms in the same way, but it does not leave "managing" unanalyzed. Instead it specifies what attribute class(es) its subject and object must belong to, and how they must be related. Specifically, the subject must be an entity (y(i)) that is an EMP, the object an IC (x) that is a MGR, and the MGR-IC must be ASsociated with the EMPloyee (AS-1). In general the translation of any verb in our theory will so specify the attribute of its subject (or the disjunction of alternatives, if any). The translation of a TV will further specify the attribute(s) of its direct object, and of a DTV of both its objects. Moreover any relationship(s) among these attributes will also be specified.

3.3. The Problems of Tense and Time

3.3.1. Intervals or States?

David Dowty in [Dowty 79] presents a discussion of a broad spectrum of semantic and syntactic issues relevant to the understanding of English, and in particular to providing a Montague-semantic analysis of these issues. In the final chapter of this book he formalizes many of the ideas he has discussed by defining a Montague fragment of English that includes such features as temporal adverbs, dative-taking verbs, a theory of word formation, and a treatment of several compound tense structures. In order to provide a semantics for this expansion of the PTQ fragment Dowty argues for the necessity of several significant extensions to the logic IL: a radically different treatment of the phenomenon of tense is one of his contributions. Because we are concerned with many of the same issues as Dowty -- in particular tenses and direct temporal references -- it seems appropriate to discuss his work and to contrast two different solutions to some of the same issues.

A major section of the book is concerned with developing a rigorous taxonomy of verbs in English based

upon several syntactic and semantic criteria. The problems with a number of different classification schemes that have been proposed over the years are discussed, in particular Vendler's scheme [Vendler 67] which divides verbs into the four categories of statives, activities, accomplishments, and achievements. Dowty judges all of these proposals by the two criteria of syntactic and semantic uniformity: can all of the verbs assigned to a given class appear in the same syntactic constructs, and are the same inferences in meaning justified for all like-classified verbs? Dowty's final taxonomy, offered with many reservations, defines eight different verb categories.

These "aspectual" verb distinctions, and particularly the semantics of the progressive tenses, lead Dowty to espouse a theory of interval semantics, earlier proposed by Bennett and Partee [Bennett 72], wherein truth conditions are given relative to an interval, rather than to a moment, of time. Unlike other proposed changes to Montague's PTQ analysis of English, this proposal causes major modifications to the most basic semantic notion of IL, and indeed of most other temporal logics that have been studied (e.g. [RescherUrquhart 71].) We are not convinced of the necessity of taking this step (indeed Dowty himself says that "it results in a system that is really too powerful for natural language semantics" [Dowty 79], p.138). Certainly from the perspective of database querying the complications that it introduces into the logic seem unnecessary.

The existence of an actual historical database as the heart of our logical model is the major constraint within which all of our work must be undertaken. This "given," which in essence already takes a stand on the semantics of the real world, stands as the major difference between Dowty's enterprise and ours. The semantic theory that we present is a theory of the semantics of English when used as a database query language for an HRDB, and not when used in "ordinary discourse," whatever that might be. If these two theories diverge, it should be neither surprising nor disturbing, and it should be of interest to compare and contrast them.

It is apparent that an HRDB is a gross abstraction of the real world: entities are represented by unique identifiers, complex relationships are reduced to simple tuples in relations, time is rather crudely represented as a set of states of gross, perhaps even somewhat amorphous, granularity. And yet in spite of these limitations these databases are found to be useful to a large and growing number of people. What kind of constraints does the abstraction of "real-world" semantics embodied in the HRDB impose upon our enterprise? Precisely this: the historical database embodies a semantics that is based upon the notion

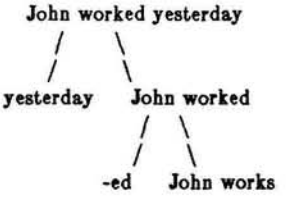
of truth with respect to a state. Every fact in the database is recorded with respect to a state which "time-stamps" it; this is interpreted as asserting that the fact is true *at that state*. If intervals come into play, they can be modelled as sets of time points, since time in our database view is discrete.

The differences that Dowty examines certainly exist; they do not, however, seem relevant in most database applications. Unlike Dowty, our analysis treats the progressive tenses synonomously with their simple counterparts:

Did John earn 30K last week? *and* **Was John earning 30K last week?**
Does Peter work for the Toy Department? *and* **Is Peter working for the Toy Department?**

Perhaps it is the case that database applications do not lend themselves to handling achievements or accomplishments, but instead record stative information. In any case it is difficult to conceive of many real database examples where distinctions of the sort that Dowty's analysis is concerned with actually make sense.

Dowty's analysis of the interaction of tenses and temporal expressions accords exactly with our own. Sentences with such interaction, such as "John worked yesterday," cannot be analyzed as resulting from two separate temporal operators (-ed and yesterday) acting on the proposition that John works, as the following example should make clear:

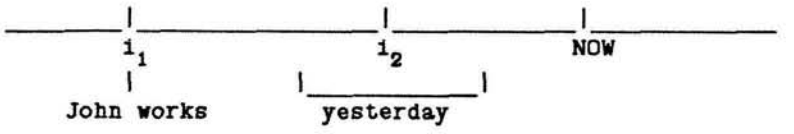


John works $\rightarrow$ work'(i)(John)
-ed $==>$ $\lambda p \exists i_1 [[i_1 < i] \wedge p(i_1)]$
John worked $==>$ $\lambda p \exists i_1 [[i_1 < i] \wedge p(i_1)](\lambda i$ work'(i)(John))
$\rightarrow$ $\exists i_1 [i_1 < i] \wedge$ work'(i_1)(John)]
yesterday $==>$ $\lambda p \exists i_2$ [yesterday'$(i_2) \wedge p(i_2)$]
John worked yesterday $==>$ $\lambda p \exists i_2$[yesterday'$(i_2) \wedge p(i_2)](\lambda i \exists i_1 [[i_1 < i] \wedge$ work'(i_1)(John)])
$\rightarrow$ $\exists i_2 \exists i_1$[yesterday'$(i_2) \wedge [i_1 < i_2] \wedge$ work'(i_1)(John)]

This analysis[2] (or the reverse which would first apply "yesterday" and then "-ed") causes the two time operators to compete with each other, placing the event in the wrong time frame. For example, the following time line is consistent with this logical analysis, but inconsistent with the intended meaning of

[2] In all translation examples we follow Partee in using a double arrow "==>" to indicate the immediate result of applying a Translation rule of the fragment, and a single arrow "$\rightarrow$" to indicate the result of any of a number of logical simplifications (principally λ-reduction.)

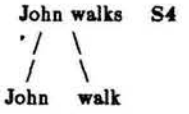
the English:



Instead the two temporal operators must be treated as operating in conjunction; the English -ed is, in a sense, semantically superfluous in the presence of the other time indicator. Thus the fragment has rules for applying tense operators, and separate rules for applying tense operators in conjunction with other temporal adverbials. These rules differ slightly from Dowty's in that we treat all temporal operators as operating on entire clauses, rather than simply on verb phrases. The next section will explore some of the reasons for this decision.

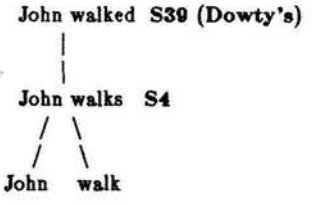
3.3.2. Sentential vs. Verb-phrasal Temporal Operators

Our analysis of tense differs from the PTQ analysis and the one in Dowty in the manner in which tense is incorporated into an English sentence. In PTQ, the rule S4 combines a Term with an IV to from a present tensed sentence:



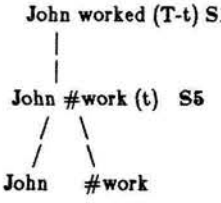
The past and future tenses are accommodated in rule S17, which similarly combines the subject and predicate to form a sentence in either of these tenses.

Dowty's analysis is somewhat different. In his fragment a sentence is always formed first by using S4; if the tense is other than present, he introduces this with an additional rule which takes the present-tensed sentence as input and forms its past-tense counterpart, as in the following example:



Extensions to the PTQ fragment have had to deal with this issue of tense and how it interacts with the other components of a sentence. We agree with Dowty's basic premise that tense is really a property of the sentence (actually, clause) as a whole. This is particularly important when, as in our fragment, there

are different kinds of sentences: declarative, WH-questions, Yes-No-questions, and When-questions. For under a straightforward extension of the PTQ treatment the number of rules would proliferate, since separate rules would be needed for each kind and tense of the sentence formed by conjoining a Term and a VP. However under Dowty's treatment, the tense rules applied after S4 in most cases must undo the syntactic work that it has done, viz. the inflection of the verb as third person singular present tense. (Semantically the treatment is the same, i.e., the untensed version denotes exactly what the present-tensed version does.) This syntactic undoing is both inelegant and computationally unattractive. For this reason, we have incorporated into QE-III the additional categories of Tensed sentences of each variety, and have modified S4 so that it creates an untensed sentence from a Term and an IV. The strings of ultimate interest in the fragment, then, are the tensed sentences (categories T-t, T-WHQ, T-YNQ, and WHENQ). The following example from QE-III illustrates this for a simple declarative sentence:



In Section 4, when we discuss further examples of tensed sentences, particularly tensed questions and when-questions, we will discuss this issue further.

3.4. Questions

3.4.1. Introduction

Despite their obvious importance as a tool for gaining knowledge of the world, both linguists and philosophers have historically considered interrogative sentences the poor relation of the declaratives, to which they have paid the bulk of their attention. Among linguists there is no generally accepted theory about the syntactic generation of English questions ([KunoRobinson 72], [Pope 76]), and philosophers and logicians have until recently given little attention to the question of questions. More recently Engdahl ([Engdahl 86]) explored the issue of constituent questions in Swedish, and proposed a semantic theory of questions similar to those of [Hamblin 73] and [Karttunen 77] which we shall discuss in Section 3.6. [GroenendijkStokhof 83] addresses the issue of the *appropriateness* of an answer in different situations, an issue outside the scope of the present work. Formal logic from its inception directed its attention to languages based upon the notion of formulas, abstractions of declarative sentences in natural languages.

Only recently have logicians begun to investigate the semantics of questions in any depth, and to develop formal languages powerful enough to express questions in order to carry out these investigations. [Hintikka 74] discusses a number of interesting linguistic and philosophical attempts to provide an analysis of questions.{

Although Montague, too, focussed his attention on a formal treatment of the syntax and semantics of declarative sentences in natural language, the framework of using a lambda-calculus and the model-theory of intensional logic, developed in PTQ, is rich enough to incorporate a view of natural language questions as well. In what seems to be his only published remark on the issue of questions he says: "In connection with imperatives and interrogatives truth and entailment conditions are of course inappropriate, and would be replaced by fulfillment conditions and a characterization of the semantic content of a correct answer" [Montague 73].

Perhaps inspired by this comment, a number of researchers have been investigating ways to incorporate a formal account of the syntax and semantics of questions within the framework of Montague semantics. [Hamblin 73], [Karttunen 77], [Bennett 77] & [Bennett 79], [HausserZaefferer 78] and [Belnap 82], are perhaps the most important of these investigations, and we will discuss their work in relation to ours in the following section. Many of the aspects of our proposal have been adapted from or influenced by the work of these researchers.

Others not working within the MS framework have also made important contributions to our understanding of the issues involved. Approaching this issue from an entirely different perspective, researchers in Artificial Intelligence (AI) have over the years developed and implemented automatic question-answering theories and systems to varying degrees of success. These have ranged from some early experimental programs [Green et al. 63] to database querying programs bound to a particular database domain ([Woods et al. 72] and [Waltz 78]) to some rather sophisticated DBQ systems today that are designed to be general and easily portable ([Harris 73], [Hendrix et al. 78].) The research behind these systems seems to share a goal common to much of the work in AI (as distinct from Cognitive Science), i.e. an interest more in getting a system to "work" than in developing a formal theory that explains its behavior.

3.4.2. Database Questions

As guidelines to help us judge any proposed theory of questions we have adopted a number of self-imposed criteria that any solution acceptable to us should meet.

1. It must fall within the general confines of Montague's framework: syntax and semantics defined in parallel, with the semantics of a phrase defined compositionally in terms of the semantics of its components.

2. The interpretation of questions should be closely analogous to the interpretation of queries in the relational database model. This means that their interpretation should be objects in the logical model which have direct analogs in the HRDM model described in Section 2.

   In the relational database model "a query is a computation upon relations to yield other relations" [Maier 83]. This is an operational view of a database query; a denotational semantics view would hold that a query denotes a relation that is its answer, and would define just how, in fact, the query so denoted. In order to provide for the closest possible parallel between the interpretation of questions in our theory and the query semantics of HRDM, we hoped to define the semantics of our English query language in just such a way, viz., such that each query would denote the relation that is its answer with respect to the database. In other words, if a query in the relational database context denoted an n-ary relation over entities (i.e., a set of n-tuples), we felt that its expression as a question in our fragment should be defined to denote a function of type $<e^n,t>$. As we shall see we were able to accomplish this easily and naturally not in the semantics, but by extending the framework of Montague Semantics to include a pragmatic component.

3. The theory should be computationally tractable. Because we are interested in developing a theory for natural language query systems that are ultimately implementable, this criterion lead us to direct our attention to solutions that within the general PTQ framework. This is because there have been successful results [Warren 79] and [Landsbergen 81] implementing parsers and semantic interpretation routines for fragments defined within this framework, and we wanted to build upon this work as much as possible. While this work does not discuss a computer implementation of its results, an extension of Warren's PTQ parser [Warren 79] to the QE-III fragment has been implemented by [Hasbrouck 82].

4. Proper treatment must be given to the interaction of questions and quantifiers. The PTQ treatment successfully accounts for multiple readings of sentences with interacting quantifiers ("A woman loves every man.") Our solution should likewise allow for all of the readings of questions involving quantified terms ("Who manages every employee?")

5. Y/N questions, WH-questions involving "who" and "what," and temporal questions ("when") should be provided for. This means that we do not treat indirect questions ("Tell me whether ..."), since these do not generally arise within the database framework and could nevertheless easily be paraphrased as direct YNQs.

6. The theory should account for multiple WH-questions (e.g. "Who sells what to whom?") as these seem indispensable in a database context.

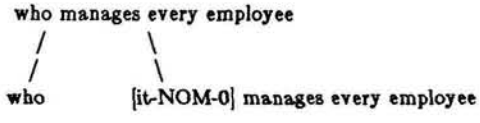
The problem of providing a correct analysis of questions that involve quantified terms is illustrated by a query like "Who manages every employee?" An analysis should only be considered adequate if it is able

to find such a query ambiguous between an interpretation of "every" as "all" and also as "each." In PTQ Montague provided a solution to the familiar problem of the multiple readings of such sentences as "A woman loves every man." Under one reading there is a single woman who (magnanimously) loves each and every man, while under the other reading there is, for each man, some woman or other who loves him. A similar problem arises with respect to the interaction between "ordinary" and question Terms, as in "Who manages every employee?"

Under one reading the questioner wishes to know what individual(s) manage all of the employees, whereas under the other reading what is wanted is really a set of ordered pairs, viz., for each employee, the set of individuals who manage him/her. Our interpretation of English questions must permit both readings, since either one is possible; the problem of disambiguating between the two is best left, as in PTQ, to a later stage that has access to domain-dependent Meaning Postulates.

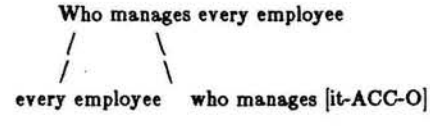
In order to get these readings, we propose making a change in the standard interpretation of the English word "every.". It is well known that this word is ambiguous -- in some cases it means "all" and in others "for each." This is precisely the ambiguity in this case, and we must provide for both readings.

The first reading, where "who" has wider scope than "every," presents no problems:



[it-NOM-O] manages every employee ==> $\forall x[EMP'(i)(x) \rightarrow MGR'(i)(x_0) \wedge manage'(i)(x_0,x)]$
who ==> $\lambda P \exists y[y(i) = u \wedge P(i)(y)]$
who manages every employee → $\exists y[y(i)=u \wedge \forall x[EMP'(i)(x) \rightarrow MGR'(i)(y) \wedge manage'(i)(y,x)]]$

The other reading requires the opposite scoping:



as well as a different meaning for "every employee." The desired reading is accomplished by allowing "every" to be ambiguous between its standard meaning "for all" and its interrogative meaning of "for each" in which it is essentially synonymous with "which."

With this treatment of "every" we obtain the desired second (and more likely) reading:

who manages [it-ACC-O] → $\exists y[y(i) = u \wedge MGR'(i)(y) \wedge EMP'(i)(x_0) \wedge manage'(i)(y,x_0)]$
every employee → $\lambda P \exists z[z(i) = v \wedge EMP'(i)(z) \wedge P(i)(z)]$

who manages every employee? → $\lambda P \exists z[z(i) = v \wedge EMP'(i)(z) \wedge P(i)(z)]$
$(\lambda i \lambda x_0 \exists y[y(i) = u \wedge MGR'(i)(y) \wedge EMP'(i)(x_0 \wedge manage'(i)(y,x_0)])$
→ $\exists z \exists y[z(i) = v \wedge EMP'(i)(z) \wedge y(i) = u \wedge MGR'(i)(y) \wedge manage'(i)(y,z)]$

which denotes a set of ordered pairs <u,v> such that u manages v.

The problem of multiple WH-questions has a rather simple solution if one is willing to restrict one's attention to questions that involve only one WH-word; it is well known, however, that multiple-WH words require a considerably more complex treatment if the semantics is to be defined compositionally as in a Montague framework it must [KunoRobinson 72]. Furthermore, within the database context a restriction to single WH-questions would be too severe a constraint -- it would limit the language to queries that return relations over only a single attribute.

We will discuss a number of different possible solutions to this issue of multiple WH-questions and ultimately adopt one as our solution. We will see, however, in the course of this presentation, that there are considerable technical difficulties in defining the semantics in such a way as to get it all to come out right for both single- and multiple-WH questions. The solution that we adopt, involving the addition of a formally specified pragmatics for the fragment, does have this property in addition to meeting our other criteria; moreover, the simplicity of our solution, as contrasted with the considerable complexity in other proposals for a question semantics, e.g. [Bennett 77] & [Bennett 79] and [HausserZaefferer 78] makes it especially attractive. However, it is clear that many researchers have found the same kinds of difficulties in extending Montague's work in the direction of interpreting questions, and that further work in this area is needed. We hope that our proposal to treat the answering of a question as a component of a formally-specified pragmatics of the language, apart from its semantics, is a step in the proper direction.

3.5. The QE-III Theory of Questions

3.5.1. Introduction

We first present a general view of the substance of our theory of the interpretation of questions and then discuss how this theory is carried out technically for the various types of questions that we consider. Our goal is a formal interpretation of questions as the set of their correct answers with respect to an index and a model (state and database.) This viewpoint is inspired by the relational database querying paradigm, wherein a query "denotes" the relation that is its answer in the current state of the database. It will be important to keep in mind the distinction between objects in a model for IL_s and objects in the

relational database model. In the relational model, particularly when dealing with the relational algebra, one tends to think of all relations as being the same kind of object. One projects and joins relations at will, since these relational operators are defined *generically*. However, models for IL_s are strongly typed: considerations of the domains and ranges of functions are of critical importance. Within IL_s, e.g., a one-place relation of individuals is a function from D_e to D_t (denoted by expressions of type <e,t>), a two-place relation of individuals a function from D_e to functions from D_e to D_t (denoted by expressions of type <e,<e,t>>), etc. Thus under our theory a question such as "Who manages Peter?" is pragmatically interpreted (in a sense to be made clear below) as an object of a completely different type from the interpretation of a question such as "Who manages whom?" Later on we will see that this theory does not fall within the mainstream of the logical theories for question semantics that have been proposed.

3.5.2. Yes-No Questions

A semantic analysis of Yes-No questions (YNQs) that meets the criteria set forth in the introduction to this section is not difficult to obtain. Since we want to interpret YNQs as either "Yes" or "No" (or equivalently T or F, or 1 or 0), they can be defined to denote objects in {0,1}. But this is just the denotation set of the corresponding declarative sentence that expresses the proposition that the YNQ asks. Thus we easily meet our criteria by providing that a YNQ denote the same proposition as that denoted by the declarative sentence from which it was derived. For example,

(3-1) John manages the shoe department.

would roughly be translated as: manage'(i)(John, Shoe Dept.)

This formula is true with respect to a state *s* just in case John manages the shoe department in that state. Our analysis of the corresponding question "Does John manage the shoe department?" provides that it is derived syntactically from "John manages the shoe department" and that semantically it denotes the same object in the model. The pragmatic interpretation of this question is represented by the formula: manage'(**now**)(John, Shoe Dept.), which in effect "questions" the model as to its truth or falsity in the same way that a YNQ questions the database for the response "yes" or "no." This analysis is provided by the following pair of syntactic and semantic rules for our fragment, and by the pragmatic rules to be introduced in Section :

S101. [YNQ Formation]

$<F_{101a}, <t>, YNQ>$ and $<F_{101b}, <t>, YNQ>$

$F_{101a}(\theta)$ = #AUX θ^* where θ^* is θ with the "first verbs" unmarked.

$F_{101b}(\theta)$ = "Is it the case that" θ

T101. $F_{101a}(\theta)$ and $F_{101b}(\theta)$ ==> θ'

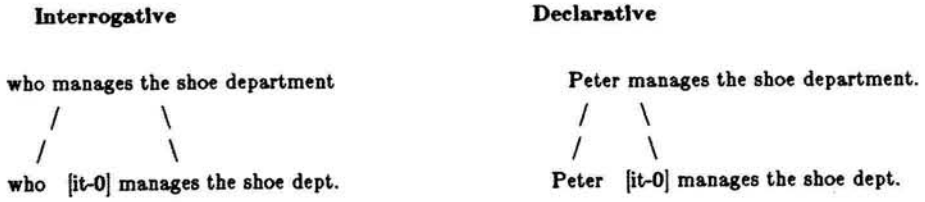
This '==>' notation is used in each translation rule that is not an instance of the general rule of function application. In this case it indicates that the translation of the expression formed by performing the operation F_{101a} (or F_{101b}) on the input string ϕ is exactly the same as the translation which has already been assigned to ϕ, which we denote with the notation ϕ'. This semantic account works, since we want the interpretation of the Yes-No question to be the same as the interpretation of the declarative sentence from which it is derived.

In what follows we examine the more difficult problem of defining compositionally a model-theoretic semantics for general WH-questions.

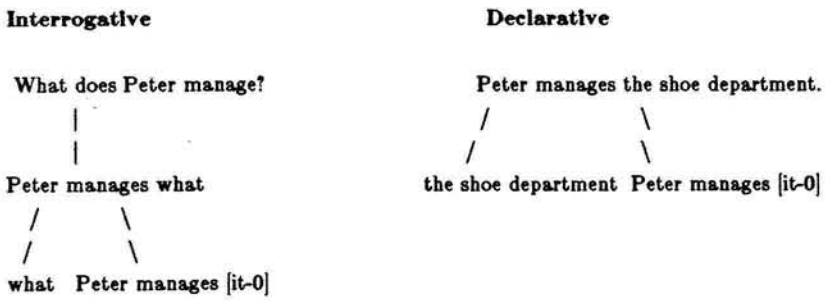
3.5.3. WH-Questions

We first present a semantic solution that does provide for a successful interpretation for questions involving only one WH-word, e.g. "Who manages Peter?" This solution has its simplicity to recommend it, but is unfortunately unable to accommodate multiple WH-questions. We then examine a number of alternative solutions to illustrate some of the many problems involved in attempting to accommodate these multiple questions.

It is an obvious linguistic fact that question words like "who," "what," "whom," etc. behave syntactically in much the same way as Terms like "Peter" or "an employee" (e.g. see [Hamblin 73].) In subject position there is virtually no difference:



while in object position there is so-called WH-Q-Movement:



Nonetheless in both cases the question word takes the role of a Term syntactically. Semantically, however, there is a difference. Whereas ordinary terms refer to an individual or a set of individuals in the model (actually in PTQ to the set of properties of these individuals but this point need not concern us here), question terms seem in some way to refer to the set of all possible individuals that, when substituted for the individual variable in a matrix, make the resulting formula true. This observation leads to the consideration that question words should denote, not sets of properties of sets of individuals, but rather functions from sets of properties to sets of individuals that have those properties. The following table makes this analogy clearer:

Term	Logical Translation	Logical Type
Peter	$\lambda P \exists x[x(i) = Peter \wedge P(i)(x)]$	$<<s,<<s,e>,t>>,t>$
an employee	$\lambda P \exists x[EMP(i)(x) \wedge P(i)(x)]$	"
who	$\lambda P \lambda u \exists x[x(i) = u \wedge P(i)(x)]$	$<<s,<<s,e>,t>>,<e,t>>$

Because of the similarity of these WH-words to ordinary Terms, both syntactically and semantically, we shall refer to them as WH-Terms. The schematic essentials of the translations of two of the above examples will show how this analysis of the semantics of WH-Terms provides the desired analysis of the WH-question.

Peter $\Longrightarrow \lambda P \exists x[x(i) = Peter \wedge P(i)(x)]$
[it-NOM-0] manages the shoe department $\Longrightarrow$ manage'(i)(x_0,shoe-dept)
Peter manages the shoe department $\Longrightarrow \lambda P \exists x[x(i) = Peter \wedge P(i)(x)](\lambda i \lambda x_0$manage'(i)($x_0$,shoe-dept))
$\rightarrow \exists x[x(i) = Peter \wedge$ manage'(i)(x,shoe-dept)
who $\Longrightarrow \lambda P \lambda u \exists x[x(i) = u \wedge P(i)(x)]$
who manages the shoe department $\Longrightarrow \lambda P \lambda u \exists x[x(i) = u \wedge P(i)(x)](\lambda i \lambda x_0$manage'(i)($x_0$,shoe-dept))
$\rightarrow \lambda u \exists x[x(i) = u \wedge$ manage'(i)(x,shoe-dept)

The first example demonstrates the PTQ-like analysis of a declarative sentence translating into a formula whose interpretation in the model with respect to a given state is a truth value. The second example provides an analysis of an interrogative sentence containing a single WH-Term, using an analogous substitution rule. We obtain an expression of type $<e,t>$ whose denotation with respect to an

index is a set of entities, viz. the set of entities who manage the shoe department in that state.

This analysis, unfortunately, cannot be generalized. Although it can also be made to provide the desired analysis for single WH-Terms in direct or indirect object position, it will not allow for multiple WH-Questions. To see why this is the case, consider what the S and T rules for the above analysis might look like:

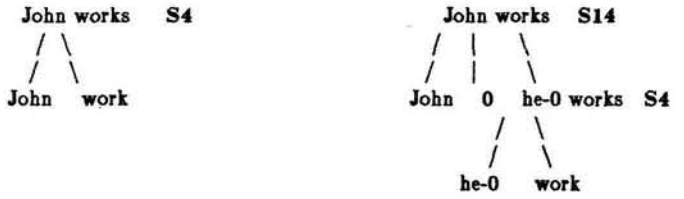
$\mathbf{S_{WH}}$. If α is a WH-Term and ϕ is a formula, then $F_{WH\text{-}n}(\alpha,\phi)$ is a WH1-?, where $F_{WH\text{-}n}(\alpha,\phi)$ would be defined as some sort of substitution of α for the first occurrence of x_n, and the appropriate pronoun for each subsequent occurrence, as in the PTQ substitution rules.

$\mathbf{T_{WH}}$. $F_{S\text{-}WH\text{-}n}(\alpha,\phi) ==> \alpha'(\lambda i \lambda x_n[\phi'])$.

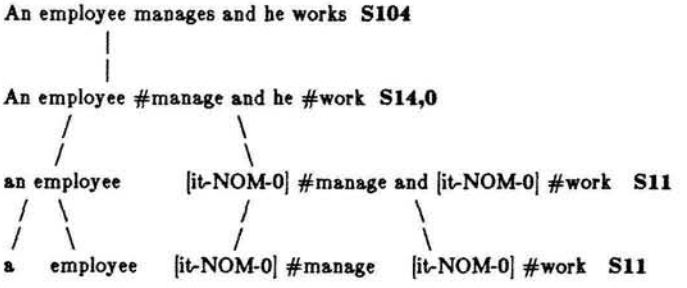
Notice that this rule, unlike the analogous substitution rules in the PTQ fragment, cannot be applied recursively to its output. This is because the PTQ rules are of the form $P_\alpha + Q_\beta ==> R_\beta$ (i.e. an expression of type α combines with an expression of type β to yield another expression of type β) whereas this rule is of the form $P_\alpha + Q_\beta ==> R_\gamma$ (i.e., the output is of a different type from either of the inputs.)

A number of alternatives present themselves at this point to allow for an analysis of multiple WH-questions within this framework. The first requires that WH-Terms have different flavors (who_0, who_1, who_2,...) depending on the meaning of the expression into which they are substituted for a free variable. The second requires subcategorizing the category Term, and substituting all Terms in for free variables at one time. The third, and the one we have adopted, achieves the same semantic effect as the rule T-WH, but in a two-stage process involving a separate pragmatic component. We will examine each of these ideas in turn. First, however, a word about substitution.

In the PTQ analysis, a Term can become a constituent part of a sentence either by directly combining with some other constituent, or indirectly by means of substitution for a free variable that has been so directly combined. For example, consider the following two PTQ-like derivations of the sentence "John works":



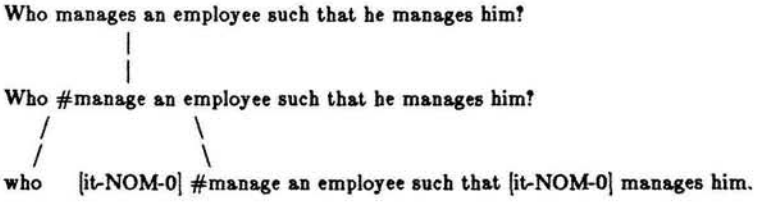
Under the semantic analysis of PTQ it turns out that these two derivations receive the same translation, and hence the same "meaning." But the substitution rules are not gratuitous. They are introduced as a theory to account for pronomial co-reference and quantifier scoping. The following example illustrates how pronomial co-reference is handled by means of one of the substitution rules:



In this derivation, the substitution rule S14,0 provides for the reading in which the same individual is the referent of the terms "an employee" and "he." The same problem of accounting for co-reference occurs in the consideration of the semantics of questions, as the following example illustrates[3] :

(3-2) Who manages an employee such that he manages him?

The PTQ theory of co-reference, extended to allow substitution of WH-Terms, is equally able to capture the fact that "who" and "him" are co-referent, as are "an employee" and "he." Under our analysis, this sentence would be derived as follows:

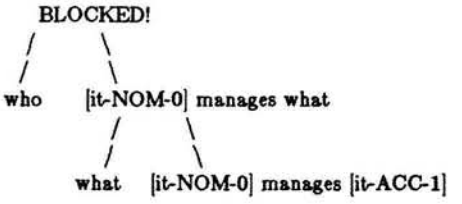


This idea of extending the PTQ theory of co-reference to the case of interrogatives is not ours. It is used in most of the work on question semantics in the Montague Grammar tradition (including [Karttunen 77],

[3]The "such that" construction for heading relative clauses is a syntactic holdover from the PTQ fragment which, because our concern is primarily semantic, we have not attempted to replace with a more sophisticated treatment. For a treatment of more "normal" English relative clauses, the reader is referred to [Cooper 79].

[Bennett 77] & [Bennett 79], and [Belnap 82]). It is that theory which we have incorporated into our fragment. Because question words in our fragment are always assumed to have the entire sentence as their scope (i.e., there are no embedded question-clauses), and because of the extensional nature of our theory as dictated by the database, question words can always be brought in indirectly by means of substitution rules. The difference in our respective treatments lies in our attempts to formalize the "meaning" given to questions.

Let us take a look now at why the analysis we have presented so far cannot be extended to multiple WH-questions. According to that analysis, the derivation of a question like "Who manages what?" is blocked after the first WH-Term is brought in:



[it-NOM-0] manages [it-ACC-1] ==> manage'(i)(x_0,x_1) **(PTQ-rules)**
what ==> $\lambda P \lambda u \exists x[x(i) = u \wedge P(i)(x)]$
[it-NOM-0] manages what ==> $\lambda \pi\ \lambda u \exists x[x(i) = u \wedge \pi(i)(x)]$
$\rightarrow\ \lambda u \exists x[x(i) = u \wedge$ manage'(i)(x_0,x)]

Syntactically the derivation is blocked because the proposed rule S-WH only allows a WH-Term to combine with a string in the category *sentence*, and under the analysis "[it-NOM-0] manages what" is not a sentence. More to the point is the semantics. "Who" denotes a function from sets of properties to sets of individuals (having those properties), and the meaning of "[it-NOM-0] manages what" is not an appropriate argument for such a function.

But suppose that the "who" which combined with formulas to form expressions denoting sets of individuals were a different function from the "who" that combined with expressions denoting sets of individuals to form expressions denoting sets of ordered pairs of individuals, etc.? Suppose, that is, that the English "who" were really a syntactic realization of a number of different meanings, who_0, who_1, etc, as follows:

who_0 combines with propositions to form a set of individuals,

who_1 combines with sets of individuals to form a set of ordered pairs, and in general,

who_i combines with sets of ordered i-tuples to form sets of ordered i+1-tuples

These different functions of the English "who" would be captured by their different translations into the logic (reflecting their interpretation as different semantic functions):

who-word	Translation	Type
who_0	$\lambda P \lambda u \exists x[x(i) = u \wedge P(i)(x)]$	$<<s,<s,e>,t>,<e,t>>$
who_1	$\lambda R \lambda v \lambda w \exists z[z(i) = v \wedge R(i)(z,w)]$	$<<s,<<s,e>,<e,>>>,<e,<e,t>>>$

With this analysis we could complete the above derivation, previously blocked, as follows:

[it-NOM-0] manages what ==> $\lambda u \exists x[x(i) = u \wedge$ manage'$(i)(x_0,x)]$
who *(as who-1)* ==> $\lambda R \lambda v \lambda w \exists z[z(i) = v \wedge R(i)(z,w)]$
who manages what ==> $\lambda R \lambda v \lambda w \exists z[z(i) = v \wedge R(i)(z,w)](\lambda i \lambda x_0 \lambda u \exists x[x(i) = u \wedge$ manage'$(i)(x_0,x)])$
→ $\lambda v \lambda w \exists z \exists x[z(i) = v \wedge x(i) = w \wedge$ manage'$(i)(z,x)]$

In theory there would be an infinite number of such (related) meanings to the word "who," one for each natural number n, and we could even give a rule for generating these meanings inductively from the single meaning of who_0. In practice (and computationally), since "ordinary" English (and even "database-ese") allows for only a small number of Terms in only a small number of places (Subject, Direct and Indirect Objects, Object of Preposition, "List..." requests, etc.) only a small number would actually ever be used in any normal English question. The S and T rules for this analysis would be something like the following:

$S_{WH\text{-}n}$. If $\alpha \in P_{WH\text{-}Term\text{-}i}$ and $\beta \in P_{t i_1 t}$ (i.e. β denotes a set of i_1 -tuples), then $F_{WH,n}(\alpha, \beta) \in P_{t i t}$, where $F_{WH,n}(\alpha, \beta)$ is the result of replacing the first occurrence of [it-CASE-n] in β with α, and replacing all subsequent occurrences of [it-CASE-n] in β with he/she/it or him/her/it, respectively, according to the gender of α and the CASE of [it-CASE-n].

$T_{Wh\text{-}n}$. $F_{WH,n}(\alpha, \beta) ==> \alpha'(\lambda i \lambda x_n \beta')$

Moreover, to account for derived WH-Terms like "which employee" in "which employee sells shoes?", we could extend this analysis to the interrogative determiners "which" and "what." This would dictate that $which_0$ combined with employee to form $[which\ employee]_0$, $which_1$ with employee to form $[which\ employee]_1$, etc., of the appropriate types.

This analysis, while inelegant, is not really so farfetched. After all, in asking "Who manages John?" "who" is in some way asking for a set of individuals, viz. those that manage John. In asking "Who manages whom?" however, rather than asking for a set of individuals, "who" is asking in conjunction with "whom" for a set of ordered pairs such that the first component manages the second component. A theory such as the above sketch would claim that English allows for these many semantic functions of

interrogative terms to be performed by the same surface words like "who."

We might also point out here a closely related alternative to this approach. Instead of having an infinite number of meanings for each WH-Term, we could suffice with one and allow an infinite number of syntactic and semantic rules for performing the substitutions. These rules would perform the necessary conversions of the meanings, not of the WH-Term, but of the sentential form into which it is being substituted. Thus, e.g., the T-WH-1 rule for combining "who" with "[it-NOM-0] manages whom" would form the following expression (where WHO* stands for the translation of "who"): $\lambda w[WHO^*[\lambda i \lambda x_0 \beta^*(w)]]$.

For example, combining "who" with "[it-NOM-0] manages what":

who $==>$ $\lambda P \lambda v \exists y[y(i) = v \wedge P(i)(y)]$
[it-NOM-0] manages what $==>$ $\lambda u \exists x[x(i) = u \wedge manage'(i)(x_0,x)]$
who manages what $==>$ $\lambda w[\lambda P \lambda v \exists y[y(i) = v \wedge P(i)(y)]]([\lambda i \lambda x_0[\lambda u \exists x[x(i) = u \wedge manage'(i)(x_0,x)](w)]])$
$\rightarrow$ $\lambda w[\lambda P \lambda v \exists y[y(i) = v \wedge P(i)(y)]]([\lambda i \lambda x_0[\exists x[x(i) = w \wedge manage'(i)(x_0,x)]]])$
$\rightarrow$ $\lambda w \lambda v \exists y \exists x[y(i) = v \wedge x(i) = w \wedge manage'(i)(y,x)]$

Notice that this rule schema essentially converts the 1-place relation denoted by one of its arguments ("[it-NOM-0] sells what" in the example) into a formula (by function application to the new individual variable w) in order to allow the single meaning of "who" to apply. Lastly, it λ-abstracts this variable w over the result in order to obtain a 2-place relation. A slightly unfortunate result of this rule is that the order of the individuals in the relation is exactly opposite from the order in which the WH-Terms were quantified in.

A second possible approach that would handle multiple WH-questions would dispense with this essentially inductive treatment of i-place questions and attack the problem all at once. Such a theory would derive all questions in the same manner, by simultaneously substituting all WH-Terms into the matrix sentence, keeping track of which terms were substituted for which variables. For example, the question "Who supplies what to which departments such that they sell shoes?" would be analyzed as follows:

"Who supplies what to which departments such that they sell shoes?"

who:0	[it-NOM-0] supplies [it-ACC-1] to [it-DAT-2]
what:1	such that [it-NOM-2] sells shoes
which department:2	

Either of these two basic theories is possible; we have rejected them both for a number of reasons. First, the use of an infinite number of meanings for WH-Terms and WH-Determiners (or an infinite number of rules schemas for their substitution) requires the same technique for each of the tense rules, and for each of the tense rules with time-adverbials, and for each of the "when"-question rules, the rules for "when"-questions with tenses, for "when"-questions with tenses and time-adverbials, etc. In other words, accepting a solution which types all questions differently depending upon what they ask for forces the inclusion of rule schemas for all of the other semantic functions that in a simple theory would operate only on one type, the type given to sentences. (Later we will discuss how the solutions of Bennett and Belnap and of Hausser and Zaefferer entail a similar rippling effect of complexity throughout the rest of the semantic theory already developed for declarative sentences.)

For an example of this effect in the theory under consideration, consider what the rule for adding past tense to a sentence would look like. (Recall our arguments for the necessity of treating tense as a property of the entire sentence.) Such a rule would have to be of the form P_α + -ED ===> Q_α where α could be the category declarative sentence (type t), 1-term question (type <e,t>), 2-term question (type <e,<e,t>>), etc. Because of the strict typing system of IL_s (and of the categorial grammar of the PTQ theory of English syntax), this would require an infinite number of such rules, one for each of the possible input categories. While such a scheme is possible, it seems to violate a concern for simplicity and elegance.

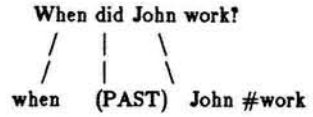
An additional problem with a theory dependent upon simultaneous substitution is a difficulty of conceiving of it in semantic terms. While the translation rules for such a theory can probably be described (they would be somewhat complicated), they strongly suggest the view that the translation rules themselves are the semantics, when in fact they are nothing more than syntactic operations on strings of logical symbols. (This is a common problem for people working with Montague Grammars, occasioned by the indirect way that the semantics for English is specified. Dowty refers to this problem [Dowty 78] when he reminds us that "the translation is a completely dispensible part of the [PTQ] theory. The 'real' semantic interpretation of an English sentence is the model-theoretic interpretation of its translation and nothing but the model-theoretic interpretation of that translation."). When examined in terms of the semantic space of functions in the model, it is not clear what simultaneous substitution in the syntax "denotes" model-theoretically.

3.5.4. Temporal Questions

"When" questions are different from any of the questions we have considered so far for three reasons. First they ask about an object of a different logical type: all of the questions we have considered have been treated as in some way referring to sets of n-tuples of individuals (of type e); "when" questions, on the other hand, refer to states (of type s).

Second, although sentences can and do make reference to more than one time ("I know that John was here"), multiple when-questions are very infrequent. In most situations "when" in English can generally only be asked once in a given sentence. "When and when ...?" does not make sense, and questions like "when did John come and when did he leave?" are really two conjoined questions.[4] We have not accounted for multiple when-questions of these sorts in our theory. An interpretation for them could be formulated in a manner analogously to our treatment of multiple WH-questions, but this would require modifying the treatment of time (modelled after Montague's) in the semantics. We will have more to say about this in our discussion of pragmatics.

Finally, when combining with WH-Terms "when" must be brought in last to have the widest scope, for essentially the same reasons that led to the recognition that tense had to have widest scope. Moreover, some account must be given of how "when" interacts with our treatment of tenses, the other major temporal indicator in the surface structure of English. The following example indicates how "when" is introduced into a sentence and "captures" the variable i in all of its free occurrences:



John #work $\rightarrow$ $EMP_*'(i)(John)$

When did John work? $\rightarrow$ $\lambda p \lambda i_1[[i_1 < i] \wedge p(i\text{-}1)](\lambda i EMP_*'(i)(John))$

$\rightarrow$ $\lambda i_1[[i_1 < i] \wedge (\lambda i EMP_*'(i)(John)(i_1)]$

$\rightarrow$ $\lambda i_1[[i_1 < i] \wedge EMP_*'(i_1)(John)]$

---

[4] Although David Warren suggests considering the following sort of exchange:

[SHE]: "There are several Fire Island ferries each day."

[HE]: "Oh, really! When do they arrive and when do they leave?"

It is natural to interpret this as a request for the set of ordered pairs $<t_1, t_2>$ representing the arrival and departure times of particular ferry runs. Other sorts of multiple when-questions that ask for a range ("Between when and when ...") seem to be of this same type.

Thus the question is interpreted as asking for the set of times in the past at which John was an employee.

3.5.5. Pragmatics or Semantics?

Most theories of question semantics, including those sketched above, and those of a number of other researchers in the Montague framework to be discussed shortly, make significant complications to the semantics of other parts of the PTQ analysis in order to incorporate these new sentences. Perhaps we are overtaxing the semantic component of our language theories, asking it to do for us more than it was intended to do. For example, most theories of question semantics have attempted to include some representation of the answer of the question as part of its denotation. Is there not something odd in a theory that holds that a question denotes its answer, especially if one has tried (or tried not) to think of "denoting" as a formal counterpart to the intuitive notion of "meaning". Yet in one form or another (denoting the set of possible answers (Hamblin), the set of correct answers (Karttunen), functions from sequences of individuals to propositions (Bennett and Belnap), and our sets of n-tuples of individuals) many researchers have been investigating ways to accomplish this in a formal semantics. The similarity between WH-Terms and unbound pronouns ("who loves whom?" versus "he loves him") suggest another approach, viz. one in which

1. the **semantic component** provides that questions *denote* as declarative sentences (with unbound pronouns) do, and

2. the **pragmatic component** provides that questions are *interpreted* as requests for their answers.

Pragmatics is the least understood branch of the tri-partite division of the study of language that [Morris 38] proposed in his theory of semiotics. This century has seen tremendous successes in the development of formal logical syntax and model-theoretic semantics, but very little in the way of formal pragmatics ([Martin 59] is an early attempt in this direction.) [Marciszewski 71], and more recently [Levinson 83] together contain a thorough analysis of the various approaches which have been taken to define the scope of linguistic pragmatics, from its origin in Morris' definition of semiotics [Morris 38] to the present day. It is obvious from these accounts that there has been less agreement as to the scope of this branch of the field. Morris originally defined pragmatics as the study of "the relation of signs to interpreters" [Morris 38]. Later, at least partly in response to Carnap's proposal that "if in an investigation explicit reference is made to the ... user of a language, then we assign it to the field of