# NATURAL LANGUAGE QUERYING OF HISTORICAL DATABASES -- THE QE-III LANGUAGE DEFINITION AND EXAMPLES

by

**James Clifford**

Information Systems Area
Graduate School of Business Administration
New York University
90 Trinity Place
New York, N.Y. 10006

May 1987

Center for Research on Information Systems
Information Systems Area
Graduate School of Business Administration
New York University

**Working Paper Series**

CRIS #164
GBA #87-90

# Table of Contents

# List of Examples

# 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** the relational model to overcome perceived shortcomings. Chief among the criticisms of RM has been its 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. Many such efforts ( [Klopprogge 81], [Clifford 82a], [Ben-Zvi 82], [CliffordWarren 83], [KlopproggeLockemann 83], [Ariav et al. 84], [Snodgrass 84], [Lum et al. 84], [Clifford 85], [SnodgrassAhn 85], [GadiaVaishnav 85]) have been presented; the HRDM ( [Clifford 85], [CliffordCroker 87]) has the advantage of being directly parallel to a formal theory of natural language.

In this paper we assume that the reader is familiar with the basic concept of an historical relational database; the HRDM that is used here is presented informally in [Clifford 85] and fully in [CliffordCroker 87]. In particular, database attributes are viewed in HRDM as **functions** from moments in time to values (in the appropriate domain), and the intensional logic $\mathrm{IL}_s$ gives us the power to speak directly about these "higher-order" objects and to incorporate 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.

[Clifford 82b] and [Clifford 87] argued that a successful formal treatment can be given to a Natural Language querying facility for an HRDB, and defined the language QE-III to show that this theory of language can serve as the formal foundation of a useable computer system for querying actual databases. QE-III is defined as a **formal** language, with syntax paired with semantics, and pragmatics defined on the two of these. Each component of the language is designed with the database application in mind. QE-III is both a simplification and an extension of the semantic theory presented in [Montague 73] and known in the literature as the PTQ fragment. Working within the framework of Montague's syntactic and semantic theory the QE-III fragment provides a treatment of English *questions* and *tenses* designed for the purposes of querying an historical database. The inclusion of a formal pragmatic component is an interesting extension to the traditional conception of a Montague Grammar. This work represents only a first step in this direction within a MS framework. The QE-III fragment 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.

This paper is intended as a supplement to [Clifford 87] for those interested in seeing the **complete** definitions of the intensional logic $\mathbb{IL}_s$ variant of Montague's $\mathbb{IL}$ and of the syntax, semantics and pragmatics of QE-III, as well as a more complete set of example derivations, translations, and pragmatic interpretations.

# 2. The Logic $\mathbb{IL}_s$

## 2.1. Introduction

As was done for the PTQ fragment, we have specified the semantics of QE-III indirectly, as follows. The set of sentences of the fragment is defined inductively from a set of Basic Expressions (words). Direct translations into an Intensional Logic are provided for each of the Basic Terms. Each of the formation rules in the inductive definition is coupled with a translation rule which specifies the translation into the logic of the output of the rule as a function of the translations of the input(s) to the rule. Thus the interpretation of any English sentence in the fragment is given by means of the model-theoretic interpretation given by the semantics of the logic to its corresponding representation in the logic.

Subsequent to the presentation of the PTQ fragment, a number of researchers (e.g., [Dowty 79], [Bennett 74], [Bennett 79], [Partee 75], [Karttunen 77], [Thomason 72a] , [Thomason 72b]) have explored various extensions to the PTQ fragment. These extensions have been motivated by the desire to provide a formal syntax and semantics to a larger set of English syntactic constructs; occasionally they have necessitated changes or extensions to the underlying logic $\mathbb{IL}$.

$\mathbb{IL}_s$ is an extension to Montague's $\mathbb{IL}$ which is motivated by the desire to provide a formal basis for the semantics of English querying of historical databases. In particular $\mathbb{IL}_s$ allows variables and constants over indices. In explaining why $\mathbb{IL}$ lacks variables (and constants) over indices, [Gallin 75] states that "$\mathbb{IL}$

was intended as a formal logic with intensional features close to those of natural language, and in natural language we do not refer explicitly to contexts of use." Thus IL contains a number of modal and tense operators (e.g., $\Box$, $\Diamond$, F, P) that allow indirect reference to indices other than the (implicitly understood) "current index," but does not allow names for specific indices. Natural language queries to an historical database, however, frequently require explicit reference to the state(s) at which the query is to be evaluated.

In HRDM, the set T consists of all of the possible states with respect to which data can be stored. An English query fragment for such a database must have Basic Expressions corresponding to such states, and a logic which is to provide the semantics for such a fragment must therefore allow constants over states in its syntax. The logic $IL_s$ is designed to serve this need. It is basically the logic $Ty_2$ presented in [Gallin 75], except that we have included more logical connectives and quantifiers as primitive symbols. Moreover we have simplified the model theory, so that an index consists solely of a time (which we call a state) rather than an ordered pair consisting of a time and a "possible world." Extending the logic to include a possible-world (or any other) component to the index is straightforward.

### 2.2. The language $IL_s$

The set of Types for $IL_s$ is the smallest set T such that:

1. e, t and s $\in$ T, and
2. if a, b $\in$ T, then $<a,b> \in$ T.

The primitive symbols of $IL_s$ are the following:

1. for each a $\in$ T, a denumerable set of variables $V_{O,a}, V_{1,a}, V_{2,a} \cdots$
2. for each a $\in$ T, a denumerable set of constants $C_{O,a}, C_{1,a}, C_{2,a} \cdots$
3. the following improper symbols: $\lambda$, (,), $=$ , $\neg$, [,],
4. $\forall$, $\exists$, $\exists!$, $\rightarrow$ , $<=>$ , , $<$, $<<$, $\wedge$, $\vee$, $\subseteq$.

The Meaningful Expressions of type a, denoted $ME_a$, are defined as follows:

1. every variable of type a belongs to $ME_a$

2. every constant of type a belongs to $ME_a$

3. if $A \in ME_{<a,b>}$ and $B \in ME_a$, then $A(B) \in ME_b$

4. if $A \in ME_b$ and x is a variable of type a, then $\lambda x A \in ME_{<a,b>}$

5. if A, B $\in ME_a$, then $[A = B] \in ME_t$

6. if $\phi, \psi \in ME_t$, and x is a variable, then $[\neg \phi]$, $[\phi \wedge \psi]$, $[\phi \vee \psi]$, $[\phi \rightarrow \psi]$, $[\phi <=> \psi]$, $\forall x\phi$, $\exists x\phi$, and $\exists! x\phi \in ME_t$

7. if A, B $\in ME_s$, then $[A < B] \in ME_t$

8. if $A \in ME_s$ and $B \in ME_{<s,t>}$, then $[A << B]$ and $[B << A] \in ME_t$

9. if A, B $\in ME_{<a,t>}$, then $[A \subseteq B] \in ME_t$

### 2.3. The Semantics of $IL_s$

A model M for the language $IL_s$ is an ordered 4-tuple $M = <E,S,<,F>$ defined as follows:

1. E is a non-empty set (the set of basic entities)

2. S is a non-empty set (the set of states)

3. $<$ is a linear ordering on S

4. F is a function which assigns to each constant $c_{n,a}$ an element in $D_a$, which is defined recursively over the set of Types T as follows:

   - $D_e = E$
   - $D_t = \{0,1\}$
   - $D_s = S$
   - $D_{<a,b>} = D_b^{\ D_a}$, *i.e., the set of all functions from $D_a$ to $D_b$*

Let **As(M)** be the set of all value assignments over M, i.e. all functions g on the set of variables of $IL_s$ such that $g(x_{n,a})$ is in $D_a$ for every variable $x_{n,a}$ of type a. If $g \in$ As(M), y is a variable of type a, and $Y \in D_a$, then **g(y|Y)** denotes the value assignment g' whose value is given as follows:

```
g'(z)  =  | Y     if z  =  y  |
          | g(z)  otherwise    |
```

Finally, we define the Denotation of the expression $A_a$ (i.e., A is of type a) with respect to a model M and a value assignment g, which we write as $Den_{M,g}(A_a)$ by the following recursion on the expression $A_a$

of $\mathrm{IL}_s$:

1. $\mathrm{Den}_{M,g}\,(x_a) \;=\; g(x_a)$

2. $\mathrm{Den}_{M,g}\,(c_{n,a}) \;=\; {}^{\bullet}F(c_{n,a})$

3. $\mathrm{Den}_{M,g}\,(A(B)) \;=\; \mathrm{Den}_{M,g}\,(A)\,(\mathrm{Den}_{M,g}\,(B))$

4.
$$\mathrm{Den}_{M,g}\,(\lambda x_a\; A_b) \;=\; \left|\begin{array}{l} \text{the function } f \text{ on } D_a \text{ whose value} \\ \text{at } X \in D_a = \mathrm{Den}_{M,g'}(A_b\,), \\ \text{where } g' = g(x_a|X) \end{array}\right|$$

5.
$$\mathrm{Den}_{M,g}\,([A = B]) \;=\; \left|\begin{array}{l} 1 \text{ if } \mathrm{Den}_{M,g}\,(A) = \mathrm{Den}_{M,g}\,(B) \\ 0 \text{ otherwise} \end{array}\right|$$

6.
$$\mathrm{Den}_{M,g}\,(\neg\phi) = \left|\begin{array}{l} 1 \text{ if } \mathrm{Den}_{M,g}\,(\phi) = 0 \\ 0 \text{ otherwise} \end{array}\right|$$

7.
$$\mathrm{Den}_{M,g}\,(\forall x_a \phi) = \left|\begin{array}{l} 1 \text{ if for every } X \in D_a,\ \mathrm{Den}_{M,g'}(\phi) = 1 \\ \quad\quad \text{where } g' = g(x_a|X) \\ 0 \quad \text{otherwise} \end{array}\right|$$

8.
$$\mathrm{Den}_{M,g}\,(\exists x_a \phi) = \left|\begin{array}{l} 1 \text{ if there exists an } X \in D_a \text{ such that} \\ \quad\quad \mathrm{Den}_{M,g'}\,(\phi) = 1, \text{ where } g' = g(x_a|X) \\ 0 \quad \text{otherwise} \end{array}\right|$$

9.
$$\mathrm{Den}_{M,g}\,(\exists! x_a \phi) = \left|\begin{array}{l} 1 \text{ if there exists a } \textbf{unique } X \in D_a \text{ with} \\ \quad\quad \mathrm{Den}_{M,g'}\,(\phi) = 1, \text{ where } g' = g(x_a|X) \\ 0 \quad \text{otherwise} \end{array}\right|$$

and similarly for $[\phi \wedge \psi]$, $[\phi \vee \psi]$, $[\phi \rightarrow \psi]$, and $[\phi \Leftarrow\!\!\!\Rightarrow \psi\,]$.

10.
$$\mathrm{Den}_{M,g}\,([A < B]) \;=\; \left|\begin{array}{l} 1 \text{ if } \mathrm{Den}_{M,g}\,(A) < \mathrm{Den}_{M,g}\,(B) \\ 0 \text{ otherwise} \end{array}\right|$$

11.
$$\mathrm{Den}_{M,g}\,([A_s \ll B_{<s,t>}\,]) = \left|\begin{array}{l} 1 \text{ if, for every } t \in D_s \text{ such that} \\ \quad \mathrm{Den}_{M,g}\,(B)(t) = 1, \text{ it is also} \\ \quad \text{the case that } \mathrm{Den}_{M,g}\,(A) < t \\ 0 \text{ otherwise} \end{array}\right|$$

12.
$$\mathrm{Den}_{M,g}\,([B_{<s,t>} \ll A_s]) = \left|\begin{array}{l} 1 \text{ if, for every } t \in D_s \text{ such that} \\ \quad \mathrm{Den}_{M,g}\,(B)(t) = 1, \text{ it is also} \\ \quad \text{the case that } t < \mathrm{Den}_{M,g}\,(A) \\ 0 \text{ otherwise} \end{array}\right|$$

13.

$$\mathrm{Den}_{M,g}\,([A \subseteq B]) = \left| \begin{array}{l} 1 \text{ if, for every } X \in D_a, \text{ whenever} \\ \quad \mathrm{Den}_{M,g}\,(A)(X) = 1 \text{ it is also} \\ \quad \text{the case that } \mathrm{Den}_{M,g}\,(B)(X) = 1 \\ 0 \text{ otherwise} \end{array} \right|$$

### 2.4. $\mathbb{IL}_s$ and $\mathbb{IL}$

The logic $\mathbb{IL}_s$ differs from $\mathbb{IL}$ primarily in treating s as a basic type along with e and t; the formula-constructing operations of the two logics are the same. It is easy to see that this makes the set of Types $T_{\mathbb{IL}_s}$ strictly larger than the set of types $T_{\mathbb{IL}}$, and that therefore the language $\mathbb{IL}$ is a proper subset of the language $\mathbb{IL}_s$. We could therefore proceed as in [Gallin 75] to define, for each expression $A_a$ of $\mathbb{IL}$, the translate of $A_a$ in $\mathbb{IL}_s$ (see [Clifford 82b]).

## 3. The QE-III Language

### 3.1. Introduction

In this section we give a formal specification of a simple English Query Fragment, QE-III, for an historical database in an imagined department store application. ( [Clifford 82b] presents a *Fragment Schema* for a QE-III type query language that could be adapted to other domains by defining the relevant vocabulary in the categories of the fragment.)

We present the language definition in three parts. First we describe the syntactic component, which consists of defining the categories of the language and the basic expressions of these categories, followed by the rules of formation. Together these constitute an inductive definition of the set of meaningful expressions of QE-III. The semantics of the language is presented next, following Montague's general procedure in PTQ. This consists of giving, for each syntactic rule, a corresponding rule of translation into the logic $\mathbb{IL}_s$, for which a direct semantic interpretation has already been specified (Section 2.) Finally, we provide a pragmatics for the language when used in the assumed context of a question-answering system. The pragmatics is presented as a set of rules that together define a function which, for any derivation tree of an expression in the language, provides what we call its pragmatic interpretation.

### 3.2. The Syntax of QE-III

## The Categories

The set of possible syntactic categories of QE-III is the smallest set CAT such that:

1. e, t, YNQ, WHQ, WHENQ, T-t, T-YNQ, T-WHQ, and Tm are in CAT, and

2. whenever A and B are in CAT, then so are A/B, A//B, and A///B.

Each of the non-empty categories of this fragment is given below, along with the following information:

1. its categorial definition, if any,

2. its corresponding type in $\mathrm{IL}_s$, and

3. the Basic Expressions of the category, if any.

The translation of the Basic Expressions into $\mathrm{IL}_s$ is given as rule T1 in the definition of the semantics of QE-III. In general, the type assignment for any category is given by the function f, defined by the following recursion on the set CAT:

1. f(e) = e

2. f(t) = f(T-t) = f(YNQ) = f(T-YNQ) = f(WHQ) = f(T-WHQ) = t

3. f(WHENQ) = f(Tm) = <s,t>

4. for all categories A and B, f(A/B) = f(A//B) = f(A///B) = <<s,f(B)>,f(A)>.

**T: Terms**
Categorial Definition: t/TV
Type: <<s,<<s,e>,t>>,t>
Basic Expressions: for each element in UD, its English name;
for each natural number i, and each CASE in
{NOM,DAT,ACC}, the "variable Terms" [it-CASE-i]

**CN: Common Nouns**
Categorial Definition: t//e
Type: <<s,e>,t>
Basic Expressions: employee, department, manager, salary, item

**IV: Intransitive Verbs**
Categorial Definition: t/e
Type: <<s,e>,t>
Basic Expressions: #work, #manage

### TV: Transitive Verbs

Categorial Definition: IV/T
Type: <<s,<<s,<<s,e>,t>>,t>>,<<s,e>,t>>
Basic Expressions: #earn, #manage, #work-for, #work-in, #have, #be

### DTV: Dative Taking Verbs

Categorial Definition: TV/T
Type: <<s,<<s,<<s,e>,t>>,t>>,<<s,<<s,<<s,e>,t>>,t>>,<<s,e>,t>>>
Basic Expressions: #supply

### S-TmADV: Sentence Time Adverbial

Categorial Definition: t//t
Type: <<s,t>,t>
Basic Expressions: sometimes, today, yesterday, 3/5/50

### VP-TmADV: Verb Phrase Time Adverbial

Categorial Definition: t///t
Type: <<s,t>,t>
Basic Expressions: always, sometimes, never

### Tm: Time

Type: <s,t>
Basic Expressions: today, yesterday, 1978, $S_1$,...

### TmPrep: Time-Phrase-Forming Preposition

Categorial Definition : (t//t)/t
Type: <<s,t>,<<s,t>,t>>
Basic Expressions: in, during, throughout, before, after

### TmConj: Time-Phrase-Forming Conjunction

Categorial Definition : S-TmADV/t
Type: <<s,t>,<<s,t>,t>>
Basic Expressions: while, before, after

### WHT: Interrogative Term

Type : <<s,<<s,e>,t>>,t>
Basic Expressions: who, whom, what

### Tm-Int: Time Interrogative

Type : <<s,t>,<s,t>>
Basic Expressions : when

### e : entity

Type: e
Basic Expressions : (none)

### t: Declarative Sentence

Type: t
Basic Expressions : (none)

**T-t: Tensed Declarative Sentence**
Type: t
Basic Expressions : (none)

**WHQ : WH-Question**
Type: t
Basic Expressions : (none)

**T-WHQ : Tensed WH-Question**
Type: t
Basic Expressions : (none)

**YNQ : Yes/No Question**
Type : t
Basic Expressions: (none)

**T-YNQ : Tensed Yes/No Question**
Type : t
Basic Expressions: (none)

**WHENQ : Tensed When Question**
Type: <s,t>
Basic Expressions: (none)

**I-DET: Interrogative Determiner**
Categorial Definition: T/CN
Type: <<s,<<s,e>,t>>,<<s,<<s,e>,t>>,t>>
Basic Expressions: which, what, how much, how many, every

**T/T: Common Noun as Term Modifier**
Type: <<s,<<s,<<s,e>,t>>,t>>,<<s,<<s,e>,t>>,t>>
Basic Expressions: (same as CN)

**DET: Determiner**
Categorial Definition: T/CN
Type: <<s,<<s,e>,t>>,<<s,<<s,e>,t>>,t>>
Basic Expressions: a, the, every

## 3.3. The Syntactic Rules of Formation

The formation rules for QE-III are given in a format similar to the format used by [Dowty 78], an amalgam of the presentation formats used by Montague in PTQ and UG. Each syntactic rule is an ordered triple of the form $<F,<I_1, I_2, \ldots I_n>,O>$, where F is the name of the structural operation performed by the rule, the $I_i$'s are the categories of the inputs to the rule, and O is the category of the output of the rule.

Rules 1 through 16 are basically the rules from the PTQ fragment adapted to $\mathbb{IL}_s$, and with some minor changes in the notational apparatus. The rules new to QE-III are presented at the end, beginning with rule number 100.

In giving the syntactic rules, we follow [Dowty 79] and describe at the beginning those common string operations that appear repeatedly, assigning them mnemonic superscripts. All other syntactic operations are described in the rule which uses them. Moreover, we subscript each syntactic operation with the number of the rule in which it appears.

$F^{I}_{n}(\alpha) = \alpha$ ***(Identity operation)***

$F^{RC}_{n}(\alpha,\beta) = \alpha\ \beta$ ***(Right Concatenation)***

$F^{LC}_{n}(\alpha,\beta) = \beta\ \alpha$ ***(Left Concatenation)***

$F^{RCA}_{n}(\alpha,\beta) = \alpha$ [it-ACC-m] if $\beta =$ [it-m]
otherwise $\alpha\ \beta$
***(Right Concatenation with Accusative Case Marking)***

$F^{RW}_{n}(\alpha,\beta) =$ the result of inserting $\beta$' after
the first word in $\alpha$, where $\beta$' is [it-ACC-m]
if $\beta$ is [it-CASE-m] (any CASE)
and $\beta$' is $\beta$ o.w. ***(Right Wrap)***

$F^{Q}_{n,m}(\alpha, \theta) =$ the result of replacing the first
occurrence of [it-CASE-m] in $\theta$ with $\alpha$,
and replacing all subsequent occurrences of [it-NOM-m],
[it-ACC-m], or [it-DAT-m] in $\theta$ with "he"/"she"/"it",
"him"/"her"/"it" or "to him"/"to her"/"to it" respectively,
according to the gender of the first basic CN or T in $\alpha$.
***(Quantification)***

Finally we note that the PTQ fragment makes use of certain auxiliary notions (the gender of a CN or a T, the he/him distinction for case markers, the proper form of verbs in several tenses, etc.) that were not rigorously defined, though they might have been. The present fragment makes use of the following additional such auxiliary notions:

case — each variable Term is marked as either uncased, or as one of NOM, ACC, or DAT. (this is a generalization of the he/him distinction that in PTQ serves to distinguish the accusative case from all others.)

plural — the plural form of each CN is required

**S1. [Basic Expressions]**

$B_A \in P_A$ for every category A.

**S2. [Determiner and Common Noun]**
$<F_2^{RC},<DET,CN>,T>$

**S3. [Relative Clauses].**
$<F_{3,n},<CN,t>,CN>.$

$F_{3,n}(\alpha, \beta) = \alpha$ "such that" $\theta^*$, where
$\theta^*$ comes from $\theta$ by replacing each occurrence of [it-NOM-n], [it-ACC-n] or [it-DAT-n] in $$\theta$ by "he"/"she"/"it", "him"/"her"/"it" or "to him"/"to her"/"to it" respectively, according to the gender of the first basic CN in $\alpha$.

**S4. [SUBJ + PRED: Untensed]**
$<F_4^{RC},<T,IV>,t>.$
If the T is a variable it is marked as being in NOM case.

**S5. [TransVerb + Direct Object]**
$<F_5^{RW},<TV,T>,IV>.$
If the T is a variable it is marked as being in ACC case.

**S6. [Preposition + Object]**
$<F_6^{RCA},<IAV/t,T>,IAV>.$

**S7. [Sentence Complement]**
$<F_7^{RC},<IV/t,t>,IV>.$

**S8. [Infinitive Complement]**
$<F_8^{RC},<IV//IV,IV>,IV>.$

**S9.** *(replaced by S103 - S105).*

**S10. [IV-Modifier]**
$<F_{10}^{LC},<IV/IV,IV>,IV>.$

**S11. [Sentence Conjunction]**
$<F_{11a},<t,t>,t>$ and $<F_{11b},<t,t>,t>.$

$F_{11a}(\theta,\psi) = \theta$ "and" $\psi$

$F_{11b}(\theta,\psi) = \theta$ "or" $\psi$

**S12. [IV-Conjunction]**
$<F_{12a},<IV,IV>,IV>$ and $<F_{12b},<IV,IV>,IV>.$

$F_{12a}(\alpha, \beta) = \alpha$ "and" $\beta$

$F_{12b}(\alpha, \beta) = \alpha$ "or" $\beta$

**S13.** [**Term Disjunction**]
$\langle F_{13}, \langle T,T\rangle, T\rangle$

$F_{13}(\alpha, \beta) = \alpha$ "or" $\beta$

**S14.** [**Quantification of Term over Declarative Sentence**]
$\langle F^{Q}_{14,n}\langle T,t\rangle,t\rangle$.

**S15.** [**Quantification of Term over CN**]
$\langle F^{Q}_{15,n}\langle T,CN\rangle,CN\rangle$

**S16.** [**Quantification of Term over IV**]
$\langle F^{Q}_{16,n}\langle T,IV\rangle,IV\rangle$

**S17.** [Negation] ***(the remaining functions of PTQ Rule 17 are handled by Rules S104 - S109.)***
$\langle F_{17}, \langle t\rangle, t\rangle$

$F_{17}(\theta) = \theta^{*}$, the result of replacing the "first verb(s)" in $\theta$ (see [Friedman 79]) by their negation.

## NEW RULES

**S100.** [**DTV + Indirect Object**]
$\langle F^{RCA}_{100}, \langle TV/T,T\rangle, TV\rangle$

If the T is a variable it is marked as being in DAT case.

**S101.** [**YNQ Formation**]
$\langle F_{101a}, \langle t\rangle, YNQ\rangle$ and $\langle F_{101b}, \langle t\rangle, YNQ\rangle$

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

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

**S102.** [**WHQ Formation**]
$\langle F_{102,n}, \langle WHT,t\rangle, WHQ\rangle$

where $F_{102,n}(\alpha,\theta)$ is defined as follows:
Let $\beta$ be the first occurrence of a variable Term [it-CASE-n] with subscript n in $\theta$ .
(1) If CASE of $\beta$ is NOM, then $F_{102,n}$is $F^{Q}_{102,n}$
(2) If CASE of $\beta$ is ACC or DAT, then $F_{102,n}(\alpha,\beta)$ is $\alpha^{*}\ \#AUX\ \theta^{*}$ or "To" $\alpha^{*}\ \#AUX\ \theta^{*}$, respectively, where
(a) $\alpha^{*}$ is "whom" if $\alpha$ is "who", and $\alpha$ otherwise,
(b) $\theta^{*}$ is $\theta$ with
(i) $\beta$ removed,
(ii) each of its "first verbs" unmarked, and
(iii) each subsequent occurrence of [it-CASE-n] replaced by "he"/"him"/"to him" according as the CASE is NOM/ACC/DAT respectively.

**S103. [WHT Quantification]**
$<F_{103,n}, <WHT,WHQ>, WHQ>$

$F_{103,n}(\alpha,\beta)$ is defined as follows:

Let $\gamma$ be the first variable Term in $\beta$. Then $F_{103,n}$ is applicable only if $\gamma$ is [it-CASE-n], (i.e., has subscript n), in which case $F_{103,n}(\alpha,\beta) = \beta^*$, where $\beta^*$ is the result of replacing $\gamma$ in $\beta$ as follows:

(a) Replace with $\alpha$ if the case of $\gamma$ is NOM, or with "to" $\alpha^*$ otherwise, where $\alpha^*$ is "whom" if $\alpha$ is "who", and $\alpha$ otherwise.

(b) Replace all subsequent variable Terms with subscript n with "he"/"she"/"it", "him"/"her"/"it", or "to him"/"to her"/"to it" respectively, according to their case and gender.

The following tense rules, S104 through S109, are perhaps best thought of as rule schemas or meta-rules. The symbol PS (for Proto-Sentence) is not really a category, but is a meta-symbol for the three possible input categories t, YNQ, and WHQ, and T-PS for the three corresponding output output categories T-t, T-YNQ, and T-WHQ, respectively. They are stated as one rule for simplicity, since the syntactic operation is the same in each case. Another way of looking at these rules is to think of the category PS as being a super-category for these other three, as in the following tree:

```
       PS
     / | \
    /  |  \
   /   |   \
  t   YNQ  WHQ
```

**S104. [Present Tense Sentence]**
$<F_{104}, <PS>, T\text{-}PS>$

$F_{104}(\theta) = \theta$, with each word of the form $\#\alpha$ replaced by its present tense form.

**S105. [Past Tense Sentence]**
$<F_{105}, <PS>, T\text{-}PS>$

$F_{105}(\theta) = \theta$, with each word of the form $\#\alpha$ replaced by its past tense form.

**S106. [Future Tense Sentence]**
$<F_{106}, <PS>, T\text{-}PS>$

$F_{106}(\theta) = \theta$ with each word of the form $\#\alpha$ replaced by its future tense form.

**S107. [Present Tense with Time-Adverbial]**

$<F_{107a},<\text{S-TmADV},t>,\text{T-t}>$

$<F_{107b},<\text{S-TmADV},\text{PS}>,\text{T-PS}>$

$<F_{107c},<\text{VP-TmADV},\text{PS}>,\text{T-PS}>$

$F_{107a}(\alpha, \theta) = \alpha\, \theta^*$

$F_{107b}(\alpha, \theta) = \theta^*\, \alpha$, where $\theta^*$ is $\theta$ with each word of the form $\#\beta$ replaced by its present tense form.

$F_{107c}(\alpha, \theta) = \theta^*$, where $\theta^*$ is the result of inserting $\alpha$ before the first word in $\theta$ of the form $\#\beta$, and then replacing each word of this form by its present tense form.

**S108. [Past Tense with Time-Adverbial]**

$<F_{108a},<\text{S-TmADV},t>,\text{T-t}>$
$<F_{108b},<\text{S-TmADV},\text{PS}>,\text{T-PS}>$
$<F_{108c},<\text{VP-TmADV},\text{PS}>,\text{T-PS}>$

$F_{108a}(\alpha, \theta) = \alpha\, \theta^*$

$F_{108b}(\alpha, \theta) = \theta^*\, \alpha$, where $\theta^*$ is $\theta$ with each word of the form $\#\beta$ replaced by its past tense form.

$F_{108c}(\alpha, \theta) = \theta^*$, where $\theta^*$ is the result of inserting $\alpha$ before the first word in $\theta$ of the form $\#\beta$, and then replacing each word of this form by its past tense form.

**S109. [Future Tense with Time-Adverbial]**

$<F_{109a},<\text{S-TmADV},t>,\text{T-t}>$
$<F_{109b},<\text{S-TmADV},\text{PS}>,\text{T-PS}>$
$<F_{109c},<\text{VP-TmADV},\text{PS}>,\text{T-PS}>$

$F_{109a}(\alpha, \theta) = \alpha\, \theta^*$

$F_{109b}(\alpha, \theta) = \theta^*\, \alpha$, where $\theta^*$ is $\theta$ with each word of the form $\#\beta$ replaced by its future tense form.

$F_{109c}(\alpha, \theta) = \theta^*$, where $\theta^*$ is the result of inserting $\alpha$ before the first word in $\theta$ of the form $\#\beta$, and then replacing each word of this form by its future tense form.

**S110. [Tensed WHENQ Formation]**

<$F_{110a}$,<Tm-Int,PS>,WHENQ>
<$F_{110b}$,<Tm-Int,PS>,WHENQ>
<$F_{110c}$,<Tm-Int,PS>,WHENQ>

$F_{110a}(\alpha,\theta) = \alpha$ "does" $\beta^*$

if $\beta$ does not begin with "To," and $\alpha$ "and" $\beta$ otherwise
where $\beta^*$ is $\beta$ with the "first verbs" unmarked.

$F_{110b}(\alpha,\theta) = \alpha$ "did" $\beta^*$

if $\beta$ does not begin with "To," and $\alpha$ "and" $\beta$ otherwise
where $\beta^*$ is $\beta$ with the "first verbs" unmarked.

$F_{110c}(\alpha,\theta) = \alpha$ "will" $\beta^*$

if $\beta$ does not begin with "To," and $\alpha$ "and" $\beta$ otherwise
where $\beta^*$ is $\beta$ with the "first verbs" unmarked.

**S111. [Tensed WHENQ Formation with Specified Time]**

<$F_{111a}$,<Tm-Int,Tm,PS>,WHENQ>
<$F_{111b}$,<Tm-Int,Tm,PS>,WHENQ>
<$F_{111c}$,<Tm-Int,Tm,PS>,WHENQ>

$F_{111a}(\alpha,\beta,\theta) = \alpha\ \beta$ "does" $\theta^*$

if $\theta$ does not begin with "To," and $\alpha\ \beta$ "and" $\theta$ otherwise
where $\theta^*$ is $\theta$ with the "first verbs" unmarked.

$F_{111b}(\alpha,\beta,\theta) = \alpha\ \beta$ "did" $\theta^*$

if $\theta$ does not begin with "To," and $\alpha\ \beta$ "and" $\theta$ otherwise
where $\theta^*$ is $\theta$ with the "first verbs" unmarked.

$F_{111c}(\alpha,\beta,\theta) = \alpha\ \beta$ "will" $\theta^*$

if $\theta$ does not begin with "to," and $\alpha\ \beta$ "and" $\theta$ otherwise
where $\theta^*$ is $\theta$ with the "first verbs" unmarked.

**S112. [TmCONJ + declarative sentence]**

<$F_{112a}$,<TmCONJ,t>,S-TmADV>
<$F_{112b}$,<TmCONJ,t>,S-TmADV>
<$F_{112c}$ ,<TmCONJ,t>,S-TmADV>

$F_{112a}\ (\alpha,\theta) = \alpha\ \theta^*$, where $\theta^*$ is $\theta$ with each word of the form $\#\beta$
replaced by its present tense form.

$F_{112b}\ (\alpha,\theta) = \alpha\ \theta^*$, where $\theta^*$ is $\theta$ with each word of the form $\#\beta$
replaced by its past tense form.

$F_{112c}\ (\alpha,\theta) = \alpha\ \theta^*$, where $\theta^*$ is $\theta$ with each word of the form $\#\beta$
replaced by its future tense form.

**S113. [TmPREP + Tm]**

$<F^{RC}_{113}, <\text{TmPREP},\text{Tm}>, \text{S-TmADV}>$

**S114. [WHT Formation]**

$<F_{114}, <\text{I-DET},\text{CN}>, \text{WHT}>$

$F_{114}(\alpha,\beta) = \alpha\ \beta^{*}$, where $\beta^{*}$ is $\beta$ if $\alpha$ is "every,"
otherwise $\beta^{*}$ is the plural form of $\beta$.

**S115. [Possessive Formation]**

$<F_{115}, <T>, \text{DET}>$

$F_{115}(\alpha) = \alpha$"'s"

**S116. [Attributive Phrase Formation]**

$<F_{116}, <\text{CN},T>, \text{CN}>$

$F_{116}(\alpha,\beta) = \alpha$ "of" $\beta$

**S117. [Role Specification for Term (I)]**

$<F_{117}, <T,\text{CN}>, T>$

$F_{117}(\alpha,\beta) = \alpha$ "as" $\beta$

**S118. [Role Specification for Term (II)]**

$<F^{RC}_{118}, <T/T,T>, T>$

## 4. The Semantics of QE-III

For the sake of clarity we have tried (within the limits of the typeset available to us) to maintain the variable-symbol conventions established by Montague in the PTQ presentation and continued (more or less) by others working within the MS framework. Moreover we have established similar conventions for the relevant new types of $IL_s$. For easy reference we give all of these conventions in the following table:

| Variable symbols | Type of variable symbol |
|---|---|
| $x, y, z, x_0, x_1, \ldots$ | $<s,e>$ : individual concepts (ICs) |
| $P, Q, Q_1, Q_2, \ldots$ | $<s,<<s,e>,t>>$ : properties of ICs |
| $p, q, q_1, q_2, \ldots$ | $<s,t>$ : propositions |
| i | s : distinguished state variable wrt which all expressions are evaluated |
| $i_1, i_2, \ldots$ | s : states |
| W | $<s,<<s,<<s,e>,t>>,t>>$ :properties of properties of ICs |

In the translation rules, $\alpha'$ and $\beta'$ (etc.) are used to represent the translations of the inputs $\alpha$ and $\beta$ to the rule. If the translation rule is given simply as "function application," this is to be understood as shorthand for the translation $\alpha'(\lambda i\beta')$ for the inputs $\alpha$ and $\beta$ . (See the presentation in Chapter III of the correspondence between $IL_s$ and IL for the discussion of the use of the distinguished time variable i and its relation to the Int operator (ˆ) in PTQ.)

### <T1>. Translations or translation schemas for Basic Expressions are as follows, by category:

### (a) T: Terms

for each English word $\alpha \in B_T$, $\alpha$==> $\lambda P\exists x[P(i)(x) \wedge x(i) = \alpha']$
for example:
Peter ==> $\lambda P\ \exists x[P(i)(x) \wedge x(i) = \text{Peter}]$

and for each variable Term [it-CASE-i], [it-CASE-i] ==> $\lambda P[P(i)(x_i)]$

### (b) CN: Common Nouns

employee ==> $\lambda x\text{EMP}_*'(i)(x(i))$
department ==> $\text{DEPT}'(i)$
manager ==> $\text{MGR}'(i)$
salary ==> $\text{SAL}'(i)$
item ==> $\lambda x\text{ITEM}_*(i)(x(i))$

### (c) IV: Intransitive Verbs

#manage ==> $\text{MGR}'(i)$
#work ==> $\lambda x\text{EMP}_*(i)(x(i)$

### (d) TV: Transitive Verbs

#manage (an employee) ==>
$\lambda W\lambda x[W(i)(\lambda i\lambda y[\text{AS-1}(y(i),x) \wedge \text{EMP}_*(i)(y(i)) \wedge \text{MGR}'(i)(x)])]$
#earn ==> $\lambda W\lambda x[W(i)(\lambda i\lambda y[\text{AS-1}(x(i),y) \wedge \text{EMP}_*(i)(x(i)) \wedge \text{SAL}'(i)(y)])]$
#sell ==> $\lambda W\lambda x[W(i)(\lambda i\lambda y[\text{REL-2}(x(i),y(i)) \wedge \text{DEPT}_*(i)(x(i)) \wedge \text{ITEM}_*(i)(y(i))])]$
#manage (a department) ==>
$\lambda W\lambda x\exists z[W(i)(\lambda i\lambda y[\text{AS-1}(z(i),x) \wedge \text{AS-1}(z(i),y) \wedge \text{EMP}_*(i)(z(i)) \wedge \text{DEPT}'(i)(y) \wedge \text{MGR}'(i)(x)])]$
#have ==> $\lambda W\lambda x[W(i)(\lambda i\lambda y\text{AS-1}(x(i),y)]$
#be ==> $\lambda W\lambda xW(i)(\lambda i\lambda y[\ x(i) = y(i)])$

### (e) DTV: Dative Taking Verbs

#supply (Company supplies item to department) ==>
$\lambda W_0\lambda W_1\lambda x[W_0(i)[\lambda i\lambda y[\text{DEPT}_*(i)(y(i)) \wedge \text{COMP}_*(i)(x(i)) \wedge$
$W_1(i)[\lambda i\lambda z[\text{ITEM}_*(i)(z(i)) \wedge \text{REL-3}(x(i),y(i),z(i))]]]]]$

### (f) S-TmADV: Sentence Time Adverbial

sometimes $==> \lambda p\exists i_1[p(i_1)]$
today $==> \lambda p\exists i_1[\text{today'}(i)(i_1) \wedge p(i_1)]$
yesterday $==> \lambda p\exists i_1[\text{yesterday'}(i)(i_1) \wedge p(i_1)]$
3/5/50 $==> \lambda p\exists i_1[3/5/50'(i_1) \wedge p(i_1)]$

N.B. today' and yesterday' are indexical constants (their denotation is relative to the current state, **now**) of type <s,<s,t>>; 3/5/50' is a non-indexical constant (its denotation is fixed, regardless of the value of **now**) of type <s,t>.

### (g) VP-TmADV: Verb Phrase Time Adverbial

always $==> \lambda p\forall i_1[p(i_1)]$
sometimes $==> \lambda p\exists i_1[p(i_1)]$
never $==> \lambda p\forall i_1[\neg p(i_1)]$

### (h) Tm: Time

if $\alpha$ is an indexical Tm (e.g. "today"), $\alpha ==> \alpha'(i)$
where $\alpha'$ is a constant of type <s,<s,t>>

otherwise if $\alpha$ is a non-indexical Tm (e.g. 1978),
$\alpha ==> \alpha'$, where $\alpha'$ is a constant of type <s,t>

### (i) TmPrep: Time-Phrase-Forming Preposition

in $==> \lambda q\lambda p[\exists i_1[q(i_1) \wedge p(i_1)]]$
during $==> \lambda q\lambda p[\exists i_1[q(i_1) \wedge p(i_1)]]$
throughout $==> \lambda q\lambda p[\forall i_1[q(i_1) \dashrightarrow p(i_1)]]$
before $==> \lambda q\lambda p[\exists i_1[[i_1 << q] \wedge p(i_1)]]$
after $==> \lambda q\lambda p[\exists i_1[[q << i_1] \wedge p(i_1)]]$

### (j) TmConj: Time-Phrase-Forming Conjunction

while $==> \lambda q\lambda p[\exists i_1[q(i_1) \wedge p(i_1)]]$
before $==>\lambda q\lambda p[\exists i_1[[i_1 << q] \wedge p(i_1)]]$
after $==> \lambda q\lambda p[\exists i_1[[q << i_1] \wedge p(i_1)]]$

### (k) WHT: Interrogative Term

$\alpha ==> \lambda P\exists y[y(i) = u \wedge P(i)(y)]$, for any $\alpha \in B_{WHT}$.

**(l) Tm-Int: Time Interrogative**

when $==> \lambda p \lambda i_1 [p(i_1)]$

**(m) I-DET: Interrogative Determiner**

$\alpha ==> \lambda Q \lambda P \exists y[y(i) = u \wedge Q(i)(y) \wedge P(i)(y)]$, for each $\alpha \in B_{\text{I-DET}}$.

**(n) T/T: Common Noun as Term Modifier**

$\alpha ==> \lambda W \lambda P\ W(i)[\lambda i \lambda y[P(i)(y) \wedge \alpha'(i)(y)]]$, if $\alpha$ refers to a role attribute

$\alpha ==> \lambda W \lambda P\ W(i)[\lambda i \lambda y[P(i)(y) \wedge \alpha_*(i)(y(i))]]$, if $\alpha$ refers to a key attribute

for any $\alpha \in B_{T/T}$.

**(o) DET: Determiner**

a $==> \lambda P \lambda Q \exists x[P(i)(x) \wedge Q(i)(x)]$
the $==> \lambda P \lambda Q \exists y[\forall x[P(i)(x) <=> x = y] \wedge Q(i)(y)]$
every $==> \lambda P \lambda Q \forall x[P(i)(x) --> Q(i)(x)]$

**T2**. function application

**T3**. $F_{3,n}(\alpha, \theta) ==> \lambda x_n(\alpha'(x_n) \wedge \theta')$.

**T4**. function application

**T5**. function application

**T6**. function application

**T7**. function application

**T8**. function application

**T9**. (replaced by T103 - T105).

**T10**. function application

**T11**. $F_{11a}(\theta, \psi) ==> [\theta \wedge \psi]$

$F_{11b}(\theta, \psi) ==> [\theta \vee \psi]$

**T12**. $F_{12a}(\alpha, \beta) ==> \lambda x[\alpha'(x) \wedge \beta'(x)]$

$F_{12b}(\alpha, \beta) ==> \lambda x[\alpha'(x) \vee \beta'(x)]$

**T13**. $F_{13}(\alpha, \beta) ==> \lambda P[\alpha'(P) \vee \beta'(P)]$

**T14**. $F^{Q}_{14,n}(\alpha, \theta) ==> \alpha'(\lambda i \lambda x_n \theta')$

**T15**. $F^{Q}_{15,n}\ (\alpha\ ,\beta) ==> \lambda y\ \alpha'(\lambda i \lambda x_{n}\ \beta'(y)])$

**T16**. $F^{Q}_{16,n}\ (\alpha,\ \beta) ==> \lambda y\ \alpha'(\lambda i \lambda x_{n}[\beta'(y)])$

**T17**. $F_{17}\ (\theta) ==> \neg\theta'$

## NEW RULES.

**T100**. function application.

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

**T102**. $F_{102,n}\ (\alpha,\theta) ==> \ \alpha'(\lambda i \lambda x_{n} \theta')$

**T103**. $F_{103,n}\ (\alpha,\beta) ==> \alpha'(\lambda i\ \lambda x_{n}\ \beta')$

**T104**. $F_{104}\ (\theta) ==> \theta'$

**T105**. $F_{105}\ (\theta) ==> \exists i_{1}[[i_{1} < i] \wedge \lambda i \theta'(i_{1})]$

**T106**. $F_{106}\ (\theta) ==> \exists i_{1}[[i < i_{1}] \wedge \lambda i \theta'(i_{1})]$

**T107**. $F_{107a}\ (\alpha,\ \theta)$, $F_{107b}\ (\alpha,\ \theta)$ and $F_{107c}\ (\alpha,\ \theta) ==> \alpha'(\lambda i \theta')$

**T108**. $F_{108a}(\alpha,\theta)$, $F_{108b}(\alpha,\theta)$, and $F_{108c}(\alpha,\theta) ==> \alpha'(\lambda i_{1}\ [[i_{1} < i] \wedge \lambda i \theta'(i_{1})])$

**T109**. $F_{109a}\ (\alpha,\ \theta)$, $F_{109b}\ (\alpha,\ \theta)$ and $F_{109c}\ (\alpha,\ \theta) ==> \alpha'(\lambda i_{1}[[i < i_{1}] \wedge \lambda i \theta'(i_{1})])$

**T110**. $F_{110a}(\alpha,\theta) ==> \lambda p \lambda i_{1}[p(i_{1})](\lambda i \theta')$

$F_{110b}(\alpha,\theta) ==> \lambda p \lambda i_{1}[[i_{1} < i] \wedge p(i_{1})]\ (\lambda i \theta')$

$F_{110c}(\alpha,\theta) ==> \lambda p \lambda i_{1}[p(i_{1})][i < i_{1}] \wedge p(i_{1})](\lambda i \theta')$

**T111**. $F_{111a}(\alpha,\beta,\theta) ==> \lambda p \lambda i_{1}[\beta'(i_{1}) \wedge p(i_{1})](\lambda i \theta')$

$F_{111b}(\alpha,\beta,\theta) ==> \lambda p \lambda i_{1}[[i_{1} < i] \wedge \beta'(i_{1}) \wedge p(i_{1})](\lambda i \theta')$

$F_{111c}(\alpha,\beta,\theta) ==> \lambda p \lambda i_{1}[[i < i_{1}] \wedge \beta'(i_{1}) \wedge p(i_{1})](\lambda i \theta')$

**T112**. $F_{112a}\ (\alpha,\theta) ==> \alpha'(\lambda i_{2}[\lambda i \theta'(i_{2})])$

$F_{112b}\ (\alpha,\theta) ==> \alpha'(\lambda i_{2}[[\lambda i \theta'(i_{2})] \wedge [i_{2} < i]])$

$F_{112c}\ (\alpha,\theta) ==> \alpha'(\lambda i_{2}[[\lambda i \theta'(i_{2})] \wedge [i < i_{2}]])$

**T113**. $F^{RC}_{113}\ (\alpha,\beta) ==> \alpha(\beta)$

**T114**. function application

**T115**. $F_{115}(\alpha) ==> \lambda W \lambda P \lambda Q \exists x[P(i)(x) \wedge Q(i)(x) \wedge W(i)[\lambda i \lambda y \text{ AS-1}(y(i),x)]] (\lambda i\ \alpha')$

**T116**. $F_{116}(\alpha\ \beta) ==> \lambda x \beta'(\lambda i\ \lambda y\ [\alpha'(x) \wedge \text{AS-1}(y(i),x)])$

**T117**. $F_{117}(\alpha,\beta) ==> \lambda P \alpha'[\lambda i \lambda y[P(i)(y) \wedge \beta'(y)]$

**T118**. function application

## 5. The Pragmatics of QE-III

The pragmatics which we give here for QE-III is a simple theory of the effects of producing an expression in that language within the assumed context of a question-answering environment. That is, we assume that a user of QE-III is using the language to produce some effect within this context, and it is this effect which we formalize as the pragmatic component of the language definition. We could, of course, have defined the pragmatics in the same manner as the semantics was defined, i.e. inductively over the syntax. However in doing so we would have seemed to be giving some status or importance to the pragmatic interpretation of expressions in every category of QE-III. Because we had no real intuition about what the **pragmatic interpretation** of, say, the expression "in 1978" represented, we decided upon a different form of the definition. Accordingly our definition provides a pragmatic interpretation for expressions in any of the several sentential categories of the language, namely T-YNQ, T-WHQ, WHENQ, and T-t. (Chapter VI contains a discussion both of some of the issues involved in our decision to present a separate pragmatic component to the formal theory of QE-III, as well as some of the considerations for the present form of this theory.)

The following preliminary definitions are needed before stating the pragmatic rules.

1. By $/\alpha\backslash$ is meant a derivation tree for the meaningful expression $\alpha$ of QE-III, as informally understood from our inductive definition of the syntax. We further assume that nodes of derivation trees are labelled with ordered triples <A,B,C> such that A is the meaningful expression derived at that node, B is its syntactic category, and C is the rule of syntax applied at that step in the derivation. For simplicity, we shall refer to component A of the root of $/\alpha\backslash$ as $\alpha$, and to the component B as CAT($/\alpha\backslash$).

2. The translation rules guarantee that corresponding to any derivation tree $/\alpha\backslash$ for $\alpha \in \mathrm{ME}_{\text{QE-III}}$ there is a unique translation into $\mathrm{IL}_s$. By $T(/\alpha\backslash)$ shall be understood this unique translation, and by $[/\alpha\backslash]_M$ the denotation of $/\alpha\backslash$ (provided indirectly via $T(/\alpha\backslash)$) with respect to the model M.

3. There are two standard ways of defining a (Tarskian) model-theoretic semantics. One is to define the notion of denotation with respect to a model M only, in which case formulas, e.g., denote the set of their satisfying variable assignments. The other, and more usual procedure (and the one that we followed in Chapter III in defining the semantics of $\mathbb{IL}_s$) is to define the denotation with respect to a model M and a variable assignment g, in which case a formula always denotes either True or False. The two notions are, for all practical purposes, equivalent. Since for the purposes of pragmatics we shall want to consider that open formulas denote the set of their satisfying variable assignments, we shall in this section refer to the notion of denotation with respect to a model M only.

4. If $[/\alpha\backslash]_M$ is a function whose domain is As(M), the set of all possible variable assignments over M, and if further $V=\{v_1, ..., v_k\}$ is a set of variables of $\mathbb{IL}_s$ , then by $\Pi_V([/\alpha\backslash]_M)$ is understood the restriction of $[/\alpha\backslash]_M$ to the domain V. Note that if $V = \theta$, then $\Pi_V([/\alpha\backslash]_M$ is defined to be just $[/\alpha\backslash]_M$.

5. If f is any function with domain As(M), then **now**(f) is the restriction of f to the domain $As_{\mathbf{now}}(M)$, where $As_{\mathbf{now}}(M) = \{g \mid g \in As(M) \text{ and } g(i) = F(\mathbf{now})\}$, that is, that subset of the possible variable assignments for M for which the distinguished time variable i is interpreted as denoting that state denoted by the constant **now**.

6. By $FV(/\alpha\backslash)$ we shall understand the set $\{i_1, i_2, ..., i_n\}$ of indices of the "variables" (expressions of the form [it-CASE-i]) occurring free in $\alpha$. This notion will not be defined rigorously here, but would be defined inductively over the structure of $/\alpha\backslash$ in the usual manner, with particular attention paid to which rules bind occurrences of variables (all of the PTQ substitution rules) and which rules leave them free (e.g., the rules that introduce WH-Terms.) This definition would be analogous to the definition of the set $FV_e$ of variables of type e occurring free in a logical expression, in particular in the expression $T(/\alpha\backslash)$. It is clear that if $FV(/\alpha\backslash) = \{i_1, ..., i_n\}$ then $FV(T(/\alpha\backslash)) = \{u_{i_1}, ..., u_{i_n}\}$. However we emphasize that $FV(/\alpha\backslash)$ is defined over the derivation tree of $\alpha$ (i.e., over the syntax of QE-III) and makes no reference to the (intermediate) translation of this tree into $\mathbb{IL}_s$.

7. Finally, if $\beta$ is a meaningful expression of $\mathbb{IL}_s$, and if the free variables of type e in $\beta$, $FV_e(\beta) = \{u_{i_1}, u_{i_2}, ..., u_{i_n}\}$, are such that $u_{i_1}, u_{i_2}, ..., u_{i_n}$ are in alphabetical order, then $LC_{FV_e}(\beta)$ is the unique expression: $\lambda u_{i_n} ... \lambda u_1 \beta$ formed by first prefixing $\beta$ with $\lambda u_{i_1}$, then prefixing $\lambda u_{i_2}$ to the result, and so on.

In order to understand the form of some of the following definitions we state the following fact (the proof follows directly from the translation rules of QE-III):

**Fact.** If $\beta$ is the translation of any meaningful expression $\alpha$ of QE-III, then the free variables of $\alpha$ are all of type e, except for the possible exception of the distinguished variable i of type s.

The rules of pragmatics which we now state constitute a definition of the pragmatic function, in a manner analogous to the way in which the translation rules constitute a translation relation. In particular they constitute a definition of the function P:

$P : /\text{QE-III}\backslash \text{ -->} M \cup \{ \text{ERROR} \}$

which assigns to any derivation tree of a meaningful expression $\alpha$ of QE-III, either an object in the model M or the distinguished symbol "ERROR" as its **pragmatic interpretation**.

P1. If $\mathrm{CAT}(/\alpha\backslash) \notin \{\text{WHENQ, T-WHQ, T-t, T-YNQ}\}$ then $P(/\alpha\backslash) = \text{ERROR}$.

P2. If $\mathrm{CAT}(/\alpha\backslash) \in \{\text{WHENQ, T-WHQ, T-t, T-YNQ}\}$ then $P(/\alpha\backslash) = \Pi_{\mathrm{FV}_e} (\mathbf{now}\ ([/\alpha\backslash]_M))$

Rule P1 ensures that only sentences are interpreted pragmatically. Rule P2 ensures that all sentences are interpreted with respect to the "current" state index, and that in the case of questions, the infinite sequences of variables that the question denotes is projected down to include only the questioned variables.

It is clear that the set of sequences given by

$\Pi_{\mathrm{FV}_e} (\mathbf{now}\ ([/\alpha\backslash]_M))$

is equivalently represented by the denotation of the expression

$\mathrm{LC}_{\mathrm{FV}_e} (\lambda i T(/\alpha\backslash)(\mathbf{now}))$

of $\mathrm{IL}_s$ with respect to M and g. P2 is therefore alternatively defined as:

$P(/\alpha\backslash) = [\mathrm{LC}_{\mathrm{FV}_e} (\lambda i T(/\alpha\backslash)\ (\mathbf{now}))]_{M,g}$.

What this alternative definition allows us to do is to utilize the semantic notion of denotation to define the pragmatic interpretation of sentences in QE-III. For it allows us to take a translation $T(/\alpha\backslash)$ of any sentence $\alpha$ and determine its pragmatic interpretation as the denotation of the expression:

$\mathrm{LC}_{\mathrm{FV}_e} (\lambda i T(/\alpha\backslash)(\mathbf{now}))$

and thus evaluate the pragmatic interpretation of $\alpha$ in terms of the semantics of $\mathrm{IL}_s$ by means of this simple syntactic transformation on $T(/\alpha\backslash)$.

# 6. Examples from the QE-III Fragment

## 6.1. Introduction

This section presents and discusses examples of the syntactic and translation rules of the QE-III fragment whose definition was given in Section 3. As we pointed out in [Clifford 87], the PTQ fragment stands essentially intact as the core of QE-III. There are, however, certain changes to this core. One major change is our use of the logic $IL_s$ as the intermediate translation language; this logic is a modification to Montague's IL, and makes explicit the "hidden" abstraction over indices that is a part of the evaluation process in Montague's PTQ analysis. In defining $IL_s$ we have already shown that we evaluate any expression $\alpha$ with respect to a state s by by forming the expression: $[\lambda i \alpha](s)$.

Moreover, in presenting the pragmatics of QE-III, we showed how the pragmatic interpretation of any sentential expression was essentially the denotation of the expression formed by λabstracting over all of the free individual variables and also evaluating with respect to **now**.

In addition to this change in the underlying logic and method of evaluation, the following additional modifications have been made to the rules of the PTQ fragment:

1. rule S4 has been modified to perform the single function of combining a Term with an IV to form a sort of proto-sentence. It no longer performs the verb inflection for 3rd person singular present tense. The entire treatment of tense and time adverbials is now performed more systematically by rules S101 through S106. (The tensing functions of S17 have therefore been totally eliminated.)

2. Montague's use of the variables $he_0$ and $him_0$ amounted to a simple technique of case marking in order to choose the appropriate personal pronoun upon substitution of a Term. We have expanded this technique somewhat, using variables of the form [it-CASE-i] where CASE ranges over {NOM,DAT,ACC} and i over the natural numbers.

3. rule S9 for combining a sentence adverbial ("Necessarily") with a sentence, has been eliminated. This is because the only sentence adverbials in QE-III are Time Adverbials which are brought in together with the tense marker in rules S104 - S106.

4. it is well known that there are problems with the PTQ treatment of conjunction and disjunction of Terms and IVs (see discussion in [Friedman 79] and [Bennett 74]). While Friedman's bracketing solution is ultimately more acceptable (both by virtue of its generality and, of particular interest, its natural correspondence to a LISP implementation), we have for simplicity of presentation adopted Bennett's simple solution of marking all Basic Verbs with a # marker which is removed when the verb is ultimately tensed. (We choose this solution because the points we wish to make have only to do with the verbs, and are easily understood

with this technique.)

For ease of understanding the translations to follow, we repeat the following table showing the types of the variables used:

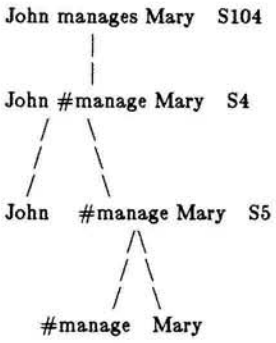
| Variable symbols | Type of variable symbol |
|---|---|
| x , y , z , $x_0$ , $x_1$ , ... | <s,e> : individual concepts (ICs) |
| P , Q , $Q_1$, $Q_2$, ... | <s,<<s,e>,t>> : properties of ICs |
| p , q , $q_1$, $q_2$, ... | <s,t> : propositions |
| i | s : distinguished state variable wrt which all expressions are evaluated |
| $i_1$ , $i_2$ ,... | s : states |
| W | <s,<<s,<<s,e>,t>>,t>> :properties of properties of ICs |

### 6.2. PTQ-like Examples from the QE-III Fragment

Before illustrating some of the added features of the QE-III database query fragment, we present some examples that fall syntactically within the range of the PTQ fragment (up to vocabulary differences) in order to contrast the way these two fragments would derive and translate the same example sentences. For example, under one analysis

**(6-1) John manages Mary**

would have the following derivation tree in QE-III:



The syntactic and translation rules illustrated in this example are **S4. [SUBJ + PRED: Untensed]**, **S5. [TransVerb + Direct Object]**, and **S104. [Present Tense Sentence]**.

Several points arise with this example. First we note that this analysis tree presents the derivational history of non-basic expressions in the language in the obvious way. Each node is labelled with a meaningful expression in QE-III; in case the expression is non-basic, it is further labelled by the syntactic rule by which it was constructed, and is given children labelled with the expressions from which it was obtained. [Montague 70] provides a more formal definition of analysis trees; it should be sufficient to point out that the language is defined in such a way that to each analysis tree (though not necessarily to each meaningful expression) there corresponds a unique translation into the intermediate logical language.

This analysis of 6-1 illustrates several departures from the corresponding PTQ analysis. First we note that the basic verb is prefixed with #, and this prefix remains even after S4 is applied to combine the Term "John" with the Intransitive Verb Phrase "#manage Mary." Second the rule S104 is new. It takes an untensed sentence as input and gives a (present) tensed sentence as output. Thus we have characterized tense as a property not of verbs but of clauses, although this property in English is realized by the inflection of the main verb of the clause. The importance of this characterization will be made clearer when we consider the interaction of tense with interrogative sentences.

This method of introducing tenses into a sentence obviates the need for undoing the English verb inflections that would be required by a method (such as in PTQ or in [Dowty 79] that *always* introduced present tense first, subject to possible subsequent modifications. [Dowty 79] (fn.5, Ch.7) makes a similar point -- though still in terms of introducing the tense via a SUBJ + PRED rule -- but does not incorporate the idea into the fragment presented there.

In a number of the PTQ rules Montague makes use of the auxiliary notions of the gender of a CN or a T, and the third person singular form of a verb. These notions are never defined with the same rigor which Montague demanded of other characteristics of his logic and grammar, presumably because he felt they were obvious and uninteresting. As in [Bennett 74] we make use of a number of similar auxiliary notions in our rules. This example points out two such notions, viz. that of the tense of a clause and the case of a variable. In our fragment a clause is either untensed or tensed, and belongs to a different category (though of the same logical type) in either case. A variable introduced into a sentence is either uncased, or one of NOM, ACC or DAT.

The translation of 6-1 corresponding to the above analysis tree is given below. In this presentation 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 (→) to indicate the result of any of a number of logical simplifications (principally λ-reduction.)

$\text{Mary} \Longrightarrow \lambda P \exists x[P(i)(x) \wedge x(i) = \text{Mary}]$
$\#\text{manage} \Longrightarrow \lambda W \lambda x[W(i)(\lambda i \lambda y[\text{AS-1}(y(i),x) \wedge \text{EMP}_{*}'(i)(y(i)) \wedge \text{MGR}'(i)(x)])]$
$\#\text{manage Mary} \Longrightarrow \lambda W \lambda x[W(i)(\lambda i \lambda y[\text{AS-1}(y(i),x) \wedge \text{EMP}_{*}'(i)(y(i)) \wedge \text{MGR}'(i)(x)])](\lambda i \lambda P \exists x[P(i)(x) \wedge x(i) = \text{Mary}])$
$\quad \rightarrow \lambda x(\lambda i \lambda P \exists x[P(i)(x) \wedge x(i) = \text{Mary}])\ (i)(\lambda i \lambda y[\text{AS-1}(y(i),x) \wedge \text{EMP}_{*}'(i)(y(i)) \wedge \text{MGR}'(i)(x)])]$
$\quad \rightarrow \lambda x \exists z[\text{AS-1}(z(i),x) \wedge \text{EMP}_{*}'(i)(z(i)) \wedge \text{MGR}'(i)(x) \wedge z(i) = \text{Mary}]$
$\text{John} \Longrightarrow \lambda P \exists y[P(i)(y) \wedge y(i) = \text{John}]$
$\text{John } \#\text{manage Mary} \Longrightarrow \lambda P \exists y[P(i)(y) \wedge y(i) = \text{John}]\ (\lambda i \lambda x \exists z[\text{AS-1}(z(i),x) \wedge \text{EMP}_{*}'(i)(z(i)) \wedge \text{MGR}'(i)(x) \wedge z(i) = \text{Mary}])$
$\quad \rightarrow \exists y \exists z[\text{AS-1}(z(i),y) \wedge \text{EMP}_{*}'(i)(z(i)) \wedge \text{MGR}'(i)(y) \wedge z(i) = \text{Mary} \wedge y(i) = \text{John}]$
$\quad \rightarrow \exists y[\text{AS-1}(\text{Mary},y) \wedge \text{EMP}_{*}'(i)(\text{Mary}) \wedge \text{MGR}'(i)(y) \wedge y(i) = \text{John}]$
$\text{John manages Mary} \Longrightarrow \exists y[\text{EMP}_{*}'(i)(\text{Mary}) \wedge \text{MGR}'(i)(y) \wedge y(i) = \text{John} \wedge \text{AS-1}(\text{Mary},y)]$

Our treatment of Proper Terms is slightly different from the PTQ treatment, in that the translations include an individual-concept variable whose extension at the state i is asserted to be the indicated individual. This is done because in HRDM all individuals of interest must be playing a role in the database, and roles can only be filled by individual concepts. Further, as we discussed in [Clifford 87], verbs are treated as objects of the same type as in PTQ, but they are analyzed in terms of the database schema.

### 6.3. Temporal Reference in QE-III

In addition to its indication by means of the tense system, temporal reference in English is also indicated by certain time adverbials (today, last year, ...) and also by prepositional phrases (in 1978, on Monday...). Care must be taken in order to analyze properly the semantics of sentences which involve an interaction between tenses and these other temporal indicators. They cannot be applied sequentially as operators to a clause, or the semantics will be incorrect. (David Dowty [Dowty 79] makes the same observation.) The following derivation for

**(6-2) Peter earned 25k in 1978.**

illustrates this aspect of QE-III:

```
Peter earned 25K in 1978     S108
     /                \
    /                  \
   /                    \
in 1978* S113   Peter #earn 25K   S4
  /  \          (derived as in example 1)
 /    \
in    1978
```

This example illustrates the following two rules: **S108. [Past Tense with Time-Adverbial]** and **S113. [TmPREP + Tm]**. The translation correctly indicates that there is some state in the past that is also in the set of states 1978 at which the present tense sentence Peter earns 25K is true:

in $==> \lambda q \lambda p[\exists i_1[q(i_1) \wedge p(i_1)]]$

1978 $==>$ 1978'

in 1978 $==> \lambda q \lambda p[\exists i_1[q(i_1) \wedge p(i_1)]]$ (1978') $\rightarrow \lambda p[\exists i_1[1978'(i_1) \wedge p(i_1)]]$

Peter earned 25K in 1978 $==>$

$$\lambda p[\exists i_1[1978'(i_1) \wedge p(i_1)]]\lambda i_2[\,[i_2 < i] \wedge \lambda i \exists y[\text{AS-1}(Peter,y) \wedge \text{EMP}_*'(i)(Peter) \wedge \text{SAL}'(i)(y) \wedge y(i) = 25K](i_2)])$$
$$\rightarrow \lambda p[\exists i_1[1978'(i_1) \wedge p(i_1)]](\lambda i_2\,[[i_2 < i] \wedge \exists y[\text{AS-1}(Peter,y) \wedge \text{EMP}_*'(i_2)(Peter) \wedge \text{SAL}'(i_2)(y) \wedge y(i_2) = 25K])$$
$$\rightarrow \exists i_1[1978'(i_1) \wedge \lambda i_2\,[[i_2 < i] \wedge \exists y[\text{AS-1}(Peter,y) \wedge \text{EMP}_*'(i_2)(Peter) \wedge \text{SAL}'(i_2)(y) \wedge y(i_2) = 25K]\,(i_1)$$
$$\rightarrow \exists i_1 \exists y[1978'(i_1) \wedge [i_1 < i] \wedge \text{EMP}_*'(i_1)(Peter) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 25K \wedge \text{AS-1}(Peter,y)]$$

If we had introduced the two temporal indicators (tense and "in 1978") separately, in either order, the resulting translations would be incorrect:

```
Peter earned 25K in 1978
   /               \
  /                 \
(PAST)    Peter #earn 25K in 1978
              /           \
             /             \
         in 1978      Peter #earn 25K
```

Peter #earn 25K $\rightarrow \exists y[\text{AS-1}(Peter,y) \wedge \text{EMP}_*'(i)(Peter) \wedge \text{SAL}'(i)(y) \wedge y(i) = 25K]$

Peter #earn 25K in 1978 $\rightarrow \exists i_1 \exists y[1978'(i_1) \wedge \text{AS-1}(Peter,y) \wedge \text{EMP}_*'(i_1)(Peter) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 25K]$

Peter earned 25K in 1978 $\rightarrow \exists i_2 \exists i_1 \exists y[[i_2 < i] \wedge 1978'(i_1) \wedge \text{EMP}_*'(i_1)(Peter) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 25K \wedge \text{AS-1}(Peter,y)]$

This places the three times $i_1$, $i_2$ and **now** on the time line as follows:

```
-------|---------|---------------------
       i2       now
```

with $i_1$ *anywhere* on the time line in 1978.

The reverse order of sequential introduction is also incorrect:

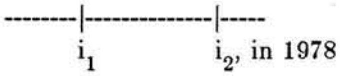
```
Peter earned 25K in 1978
    /            \
   /              \
in 1978     Peter earned 25K
              /           \
             /             \
         (PAST)      Peter #earn 25K
```

Peter #earn 25K → $\exists y[\text{AS-1}(\text{Peter},y) \wedge \text{EMP}_*'(i)(\text{Peter}) \wedge \text{SAL}'(i)(y) \wedge y(i) = 25K]$

Peter earned 25K → $\exists i_1 \exists y[[i_1 < i] \wedge \text{AS-1}(\text{Peter},y) \wedge \text{EMP}_*'(i_1)(\text{Peter}) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 25K]$

Peter earned 25K in 1978 → $\exists i_2 \exists i_1 \exists y[1978'(i_2) \wedge [i_1 < i_2] \wedge \text{EMP}_*'(i_1)(\text{Peter}) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 25K \wedge \text{AS-1}(\text{Peter},y)$

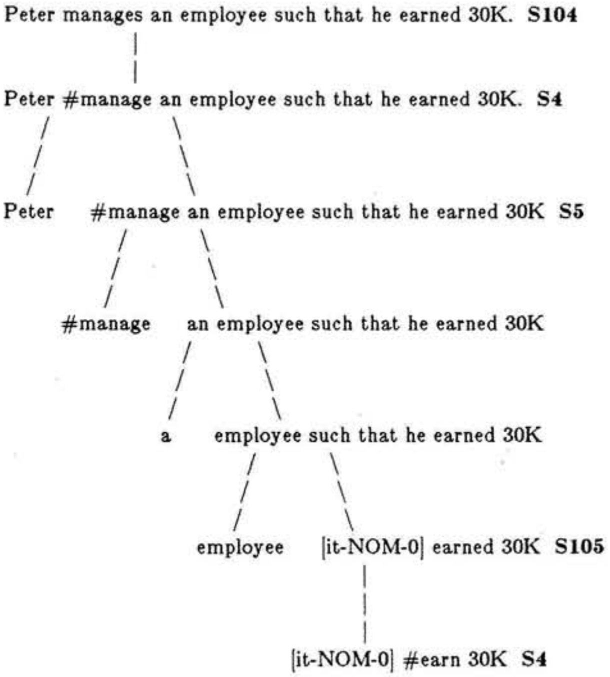
Here the two times are located as follows:



The properties of Peter are asserted to be true in state $i_1$, but $i_1$ may or may not be in 1978, and may or may not be in the past (with respect to **now**.) Only the simultaneous introduction of these temporal operators provides the correct translation.

The following example illustrates how tense is treated as a property of clauses in compound sentences, and how these tenses are independent of one another. The example also illustrates how relative clauses are maintained in the QE-III fragment:

**(6-3) Peter manages an employee such that he earned 30K.**


Peter manages an employee such that he earned 30K. **S104**
Peter #manage an employee such that he earned 30K. **S4**
Peter
#manage an employee such that he earned 30K **S5**
#manage
an employee such that he earned 30K
a
employee such that he earned 30K
employee
[it-NOM-0] earned 30K **S105**
[it-NOM-0] #earn 30K **S4**

This example illustrates the rule **S105. [Past Tense Sentence]** which introduces the past tense in a manner analogous to S104's introduction of the present. (The future is introduced by the comparable rule S106.)

Furthermore, we have dispensed with Montague's treatment of the inflection of pronouns via the technique of the variables $he_i$, $him_i$, etc., and incorporated a more general treatment that uses [it-CASE-i], where CASE is any one of NOMinative, DATive, or ACCUsative. This treatment is both more general and more easily extendible to other cases.

The translation for this analysis tree is as follows:

[it-NOM-0] earned 30K → $\exists i_1 \exists y[1978'(i_1) \wedge [i_1 < i] \wedge \text{AS-1}(x_0(i),y) \wedge \text{EMP}_*'(i_1)(x_0(i)) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 30\text{K}]$

employee ==> $\lambda x \text{EMP}_*'(i)(x(i))$

an employee such that he earned 30K →

$\lambda Q \exists x \exists i_1 \exists y[\text{EMP}_*'(i)(x(i)) \wedge 1978'(i_1) \wedge [i_1 < i] \wedge \text{AS-1}(x(i),y) \wedge \text{EMP}_*'(i_1)(x(i)) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 30\text{K} \wedge Q(i)(x)]$

#manage ==> $\lambda W \lambda x[W(i)(\lambda i \lambda y[\text{AS-1}(y(i),x) \wedge \text{EMP}_*'(i)(y(i)) \wedge \text{MGR}'(i)(x)])]$

#manage an employee such that he earned 30K →

$\lambda z \exists x \exists i_1 \exists y[\text{EMP}_*'(i)(x(i)) \wedge 1978'(i_1) \wedge [i_1 < i] \wedge \text{AS-1}(x(i),y) \wedge \text{EMP}_*'(i_1)(x(i)) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 30\text{K} \wedge \text{AS-1}(x(i),z) \wedge \text{EMP}_*'(i)(x(i)) \wedge \text{MGR}'(i)(z)]$

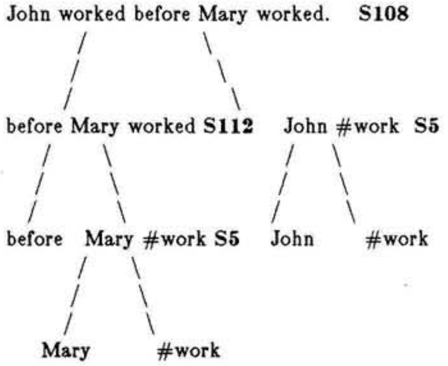
Peter manages an employee such that he earned 30K →

$\exists w \exists x \exists y \exists i_1[\text{EMP}_*'(i)(x(i)) \wedge \text{MGR}'(i)(w) \wedge w(i) = \text{Peter} \wedge \text{AS-1}(x(i),w) \wedge \text{EMP}_*'(i_1)(x(i)) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 30\text{K} \wedge 1978'(i_1) \wedge [i_1 < i] \wedge \text{AS-1}(x(i),w)]$

A final example involving tense and a temporal modifier illustrates how propositions can be treated in almost the same way as time constants for denoting sets of states. The sentence

**(6-4) John worked before Mary worked.**

is analyzed as asserting that there was some state S1 before **now** at which John worked, and that S1 was also before some other state S2 before **now** at which Mary worked. An analysis of 6-4 is given by the following tree:



Rule **S112. [TmCONJ + declarative sentence]** allows the formation of a time-adverbial phrase from a preposition, a tense, and a sentence:

The translation proceeds as follows:

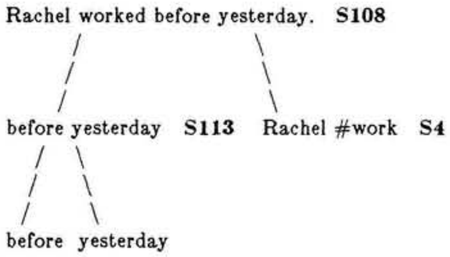
Mary #work → $\text{EMP}_*'(i)(\text{Mary})$

before ==> $\lambda q \lambda p[\exists i_1[[i_1 << q] \wedge p(i_1)]]$

before Mary worked → $\lambda p \exists i_1 [[i_1 << (\lambda i_2 \, EMP_*'(i_2)(Mary))] \wedge [i_2 < i] \wedge p(i_1)]$
John #work → $EMP_*'(i)(John)$
John worked before Mary worked → $\exists i_1 [[i_1 << (\lambda i_2 EMP_*'(i_2)(Mary))] \wedge [i_2 < i] \wedge [i_1 < i] \wedge EMP_*'(i_1)(John)]$

Similarly we can combine simple time expressions with prepositions to form temporal adverbials, as in this analysis of

**(6-5) Rachel worked before yesterday.**



The new rule that defines this is **S113.** **[TmPREP + Tm]**.

This example translates as follows:

Rachel #work → $EMP_*'(i)(Rachel)$
before ==> $\lambda q \lambda p [\exists i_1 [[i_1 << q] \wedge p(i_1)]]$
yesterday ==> $yesterday'(i)$
before yesterday → $\lambda p [\exists i_1 [[i_1 << yesterday'(i)] \wedge p(i_1)]]$
Rachel worked before yesterday → $\exists i_1 [[i_1 << yesterday'(i)] \wedge [i_1 < i] \wedge EMP_*'(i_1) \, (Rachel)]$

Notice that there are two restrictions placed upon when the state $i_1$ can occur in time:

**(1)** $[i_1 << yesterday'(i)]$ because of "before yesterday," and

**(2)** $[i_1 < i]$ because of the past tense.

Since a time before yesterday must be before **now** (by the meaning of "yesterday"), a Meaning Postulate for words such as "yesterday" might well be in order here to remove this redundancy and reduce the final translation to:

$\exists i_1 [[i_1 << yesterday'(i)] \wedge EMP_*'(i_1)(Rachel)]$

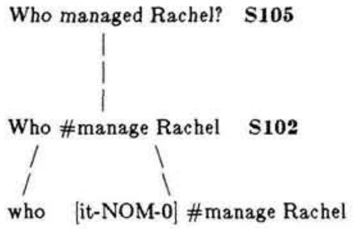
We now proceed to discuss the other additional rules of the QE-III fragment. These rules either form expressions that have particular relevance to the database realm (possessives, role specifications, etc.) or form interrogative sentences. We will look first at the questions. A discussion of some of the considerations involved in the framing of these rules for database querying purposes was given in Chapter VI.

### 6.4. Questions in QE-III

Consider the following query:

**(6-6) Who managed Rachel?**

derived as in the following tree:



The rule that introduces a WH-interrogative into a declarative sentence is **S102. [WHQ Formation]**.

The translation is a follows:

[it-NOM-0] #manage Rachel → *(as above)* $[\mathrm{AS\text{-}1}(\mathrm{Rachel},x_0) \wedge \mathrm{EMP}_*'(i)(\mathrm{Rachel}) \wedge \mathrm{MGR}'(i)(x_0)]$

who $==> \lambda P \exists y[y(i) = u \wedge P(i)(y)]$

Who #manage Rachel? $==> \lambda P \exists y[y(i) = u \wedge P(i)(y)]\ (\lambda i \lambda x_0[\mathrm{AS\text{-}1}(\mathrm{Rachel},x_0) \wedge \mathrm{EMP}_*'(i)(\mathrm{Rachel}) \wedge \mathrm{MGR}'(i)(x_0)])$

$\rightarrow \exists y[y(i) = u \wedge \mathrm{AS\text{-}1}(\mathrm{Rachel},y) \wedge \mathrm{EMP}_*'(i)(\mathrm{Rachel}) \wedge \mathrm{MGR}'(i)(y)])$

Who managed Rachel? $==> \exists i_1[[i_1 < i] \wedge \lambda i \exists y[y(i) = u \wedge \mathrm{AS\text{-}1}(\mathrm{Rachel},y) \wedge \mathrm{EMP}_*'(i)(\mathrm{Rachel}) \wedge \mathrm{MGR}'(i)(y)]\ (i_1))$

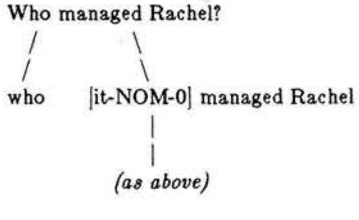
$\rightarrow \exists i_1 \exists y[[i_1 < i] \wedge \mathrm{EMP}_*'(i_1)(\mathrm{Rachel}) \wedge \mathrm{MGR}'(i_1)(y) \wedge y(i_1) = u \wedge \mathrm{AS\text{-}1}(\mathrm{Rachel},y)]$

Recall that the pragmatics provides a representation for the answer to questions, and that the pragmatic interpretation of this query is denoted by the expression

$\lambda u\ \exists i_1 \exists y[[i_1 < \mathbf{now}] \wedge \mathrm{EMP}_*'(i_1)(\mathrm{Rachel}) \wedge \mathrm{MGR}'(i_1)(y) \wedge y(i_1) = u \wedge \mathrm{AS\text{-}1}(\mathrm{Rachel},y)]$

formed by binding all free occurrences of the variable "i" to the constant **now**, and λ-abstracting over all of the free individual variables.

This example illustrates why the tense must be considered a property of the entire clause, rather than just of the verb phrase, if the semantics of the question is to come out right. For suppose instead that we derived 6-6 as follows:



The translation would proceed:

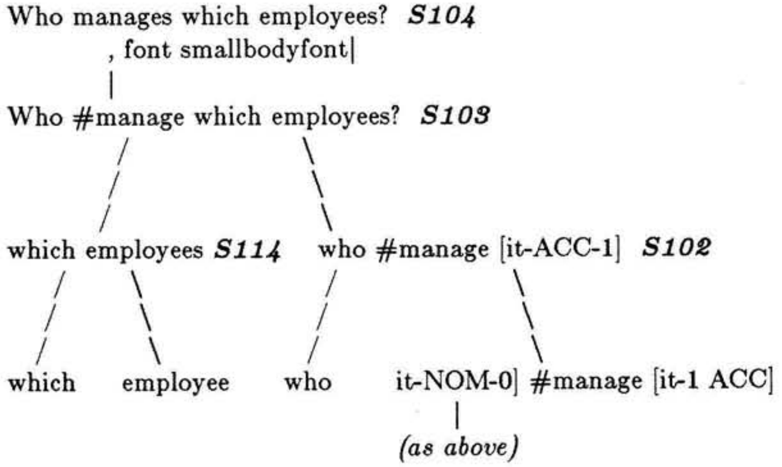
[it-NOM-0] managed Rachel $==> \exists i_1[[i_1 < i] \wedge \mathrm{EMP}_*'(i_1)(\mathrm{Rachel}) \wedge \mathrm{MGR}'(i_1)(x_0) \wedge \mathrm{AS\text{-}1}(\mathrm{Rachel},x_0)]$

who managed Rachel? $==> \lambda P \exists y[y(i) = u \wedge P(i)(y)](\lambda i \lambda x_0 \exists i_1 [[i_1 < i] \wedge EMP_*'(i_1)(Rachel) \wedge MGR'(i_1)(x_0) \wedge AS\text{-}1(Rachel,x_0)])$
$\rightarrow \exists y \exists i_1 [y(i) = u \wedge [i_1 < i] \wedge EMP_*'(i_1)(Rachel) \wedge MGR'(i_1)(y) \wedge AS\text{-}1(Rachel,y)]$

The problem with this translation is that the manager-IC y is not "tensed" properly. When evaluated, this query will return the set of individuals u who are the extension of Rachel's manager-IC, not at some time in the past, but **now**. Because "who" has wider scope in this derivation, the past tense operator could not capture the free i of the translation of "who." The question, under our treatment, is correctly analyzed as **Who (past) managed (past) Rachel?** rather than as **Who (*now*) managed (past) Rachel?** In order to get this reading, tenses (and tenses + TmADVerbials) must be brought in last over all clause, including interrogative sentences.

Interrogative Terms (WHT's) can also be derived from common nouns and the interrogative determiners such as "which," as seen in the following example which also illustrates the derivation of a multiple WH-question:

**(6-7) Who manages which employees?**


Who manages which employees? *S104*
, font smallbodyfont]
Who #manage which employees? *S103*
which employees *S114*
who #manage [it-ACC-1] *S102*
which
employee
who
it-NOM-0] #manage [it-1 ACC]
*(as above)*

Two rules are illustrated in this example. **S114. [WHT Formation]** creates an interrogative term by combining an interrogative determiner with a common noun, and **S103. [WHT Quantification]** forms multiple WH-questions.

The translation proceeds as follows:

[it-NOM-0] #manage [it-ACC-1] → *(as above)* $AS\text{-}1(x_1 (i),x_0) \wedge EMP_*'(i)(x_1 (i)) \wedge MGR'(i)(x_0)$
who #manage [it-ACC-1] → *(as above)* $\exists y[y(i) = u_1 \wedge AS\text{-}1(x_1 (i),y) \wedge EMP_*'(i)(x_1 (i)) \wedge MGR'(i)(y)]$
which $==> \lambda Q \lambda P \exists z[z(i) = u_2 \wedge Q(i)(z) \wedge P(i)(z)]$

employee ==> $\lambda x EMP_*'(i)(x(i))$

which employees ==> $\lambda Q\lambda P\exists z[z(i) = u_2 \wedge Q(i)(z) \wedge P(i)(z)]\ (\lambda i\lambda x EMP_*'(i)(x(i)))$

→ $\lambda P\exists z[z(i) = u_2 \wedge EMP_*'(i)(z(i)) \wedge P(i)(z)]$

Who #manage which employees? ==>

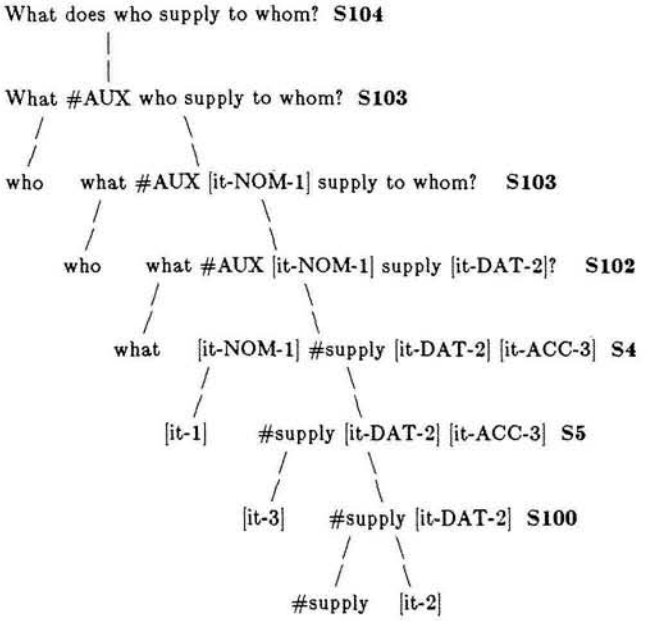
$\lambda P\ \exists z[z(i) = u_2 \wedge EMP_*'(i)(z(i)) \wedge P(i)(z)]\ (\lambda i\lambda x_1\ \exists y[y(i) = u_1 \wedge AS\text{-}1(x_1(i),y) \wedge EMP_*'(i)(x_1(i)) \wedge MGR'(i)(y)])$

→ $\exists y[EMP_*'(i)(u_2) \wedge y(i) = u_1 \wedge AS\text{-}1(u_2,y) \wedge MGR'(i)(y)]$

Who manages which employees ==> $\exists y[\ EMP_*'(i)(u_2) \wedge MGR'(i)(y) \wedge y(i) = u_1 \wedge AS\text{-}1(u_2,y)]$

The next example illustrates a 3-Term interrogative:

### (6-8) What does who supply to whom?


What does who supply to whom? S104
What #AUX who supply to whom? S103
who
what #AUX [it-NOM-1] supply to whom? S103
who
what #AUX [it-NOM-1] supply [it-DAT-2]? S102
what
[it-NOM-1] #supply [it-DAT-2] [it-ACC-3] S4
[it-1]
#supply [it-DAT-2] [it-ACC-3] S5
[it-3]
#supply [it-DAT-2] S100
#supply
[it-2]

This example uses the three-place verb "#supply" and a rule for combining such a verb with an indirect object to form a two-place verb. Rule **S100. [DTV + Indirect Object]** is essentially taken from [Dowty 79], and is a simple extension of the two-place case.

#supply ==> $\lambda W_0\lambda W_1\lambda x[W_0(i)[\lambda i\lambda y[DEPT_*'(i)(y(i)) \wedge COMP_*'(i)(x(i)) \wedge W_1(i)[\lambda i\lambda z[ITEM_*'(i)(z(i)) \wedge REL\text{-}3(x(i),y(i),z(i))]]]]]$

[1]#supply [it-DAT-2] ==>

$\lambda W_0\lambda W_1\lambda x[W_0(i)[\lambda i\lambda y[DEPT_*'(i)(y(i)) \wedge COMP_*'(i)(x(i)) \wedge W_1(i)[\lambda i\lambda z[ITEM_*'(i)(z(i)) \wedge REL\text{-}3(x(i),y(i),z(i))]]]]](\lambda i\lambda P(P(i)(x_2)))$

→ $\lambda W_1\lambda x[\lambda P(P(i)(x_2)[\lambda i\lambda y[DEPT_*'(i)(y(i)) \wedge COMP_*'(i)(x(i)) \wedge W_1(i)[\lambda i\lambda z[ITEM_*'(i)(z(i)) \wedge REL\text{-}3(x(i),y(i),z(i))]]]]]$

→ $\lambda W_1\ \lambda x[[DEPT_*'(i)(x_2(i)) \wedge COMP_*'(i)(x(i)) \wedge W_1(i)[\lambda i\lambda z[ITEM_*'(i)(z(i)) \wedge REL\text{-}3(x(i),x_2(i),z(i))]]]]$

#supply [it-DAT-2] [it-ACC-3] ==>

$\lambda W_1\lambda x[[DEPT_*'(i)(x_2(i)) \wedge COMP_*'(i)(x(i)) \wedge W_1(i)[\lambda i\lambda z\ [ITEM_*'(i)(z(i)) \wedge REL\text{-}3(x(i),x_2(i),z(i))]]]](\lambda i\lambda P(P(i)(x_3)))$

→ $\lambda x[[DEPT_*'(i)(x_2(i)) \wedge COMP_*'(i)(x(i)) \wedge \lambda P(P(i)(x_3))\ [\lambda i\lambda z[ITEM_*'(i)(z(i)) \wedge REL\text{-}3(x(i),x_2(i),z(i))]]]]$

→ $\lambda x[DEPT_*'(i)(x_2(i)) \wedge COMP_*'(i)(x(i)) \wedge ITEM_*'(i)(x_3(i)) \wedge REL\text{-}3(x(i),x_2(i),x_3(i))]$

[it-NOM-1] #supply [it-DAT-2] [it-ACC-3] → *(as above)*

$[DEPT_*'(i)(x_2(i)) \wedge COMP_*'(i)(x_1(i)) \wedge ITEM_*'(i)(x_3(i)) \wedge REL\text{-}3(x_1(i),x_2(i),x_3(i))]$

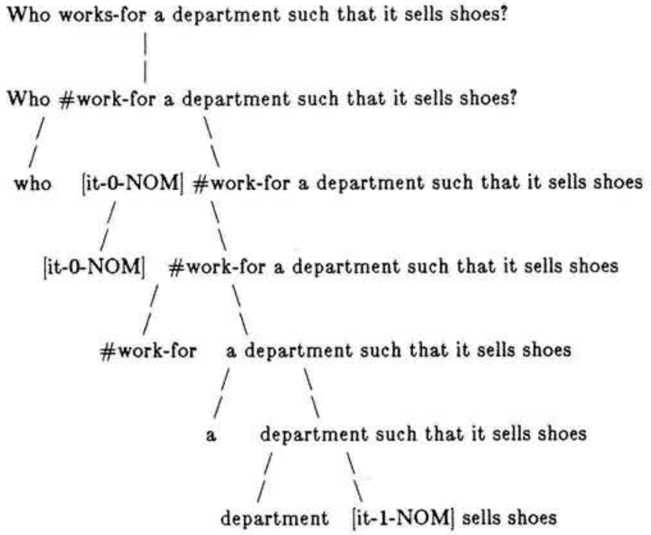
What #AUX [it-NOM-1] supply [it-DAT-2]? → $[DEPT_*'(i)(x_2(i)) \wedge COMP_*'(i)(x_1(i)) \wedge ITEM_*'(i)(u_1) \wedge REL\text{-}3(x_1(i),x_2(i),u_1)]$

[1] REL-3 indicates that there is a 3-ary relationship among the indicated three individuals.

What #AUX [it-NOM-1] supply to whom? → $[\text{DEPT}_*'(i)(u_2) \wedge \text{COMP}_*'(i)(x_1(i)) \wedge \text{ITEM}_*'(i)(u_1) \wedge \text{REL-3}(x_1(i),u_2,u_1)]$
What #AUX who supply to whom? → $\text{DEPT}_*'(i)(u_2) \wedge \text{COMP}_*'(i)(u_3) \wedge \text{ITEM}_*'(i)(u_1) \wedge \text{REL-3}(u_3,u_2,u_1)$
What does who supply to whom → $\text{COMP}_*'(i)(u_3) \wedge \text{DEPT}_*'(i)(u_2) \wedge \text{ITEM}_*'(i)(u_1) \wedge \text{REL-3}(u_3,u_2,u_1)$

The next example illustrates a more complicated question that requires, in terms of the database representation, a "join" of two relations:

**(6-9) Who works for a department such that it sells shoes?**


Who works-for a department such that it sells shoes?
Who #work-for a department such that it sells shoes?
who
[it-0-NOM] #work-for a department such that it sells shoes
[it-0-NOM]
#work-for a department such that it sells shoes
#work-for
a department such that it sells shoes
a
department such that it sells shoes
department
[it-1-NOM] sells shoes

It is translated as follows:

[it-1-NOM] sells shoes ==> $\text{DEPT}_*'(i)(x_1\ (i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x_1(i),\text{Shoes})$
department such that it sells shoes ==> $\lambda x_1\ (\text{DEPT}_*'(i)(x_1(i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x_1(i),\text{Shoes}))$
a department such that it sells shoes ==>
$\lambda P \lambda Q \exists x[P(i)(x) \wedge Q(i)(x)]\ (\lambda i \lambda x_1(\text{DEPT}_*'(i)(x_1(i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x_1(i),\text{Shoes})))$
$\rightarrow \lambda Q\ \exists x[\text{DEPT}_*'(i)(x(i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x(i),\text{Shoes}) \wedge Q(i)(x)]$
#work-for ==> $\lambda W \lambda z[W(i)(\lambda i \lambda y[\text{AS-1}(z(i),y) \wedge \text{EMP}_*'(i)(z(i)) \wedge \text{DEPT}'(i)(y)])]$
#work-for a department such that it sells shoes ==>
$\lambda W \lambda z[W(i)(\lambda i \lambda y\ [\text{AS-1}(z(i),y) \wedge \text{EMP}_*'(i)(z(i)) \wedge \text{DEPT}'(i)(y)])](\lambda i \lambda Q \exists x[\text{DEPT}_*'(i)(x(i)) \wedge$
$\text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x(i),\text{Shoes}) \wedge Q(i)(x)])$
$\rightarrow \lambda z[\lambda Q \exists x[\text{DEPT}_*'(i)(x(i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x(i),\text{Shoes}) \wedge Q(i)(x)](\lambda i \lambda y[\text{AS-1}(z(i),y) \wedge \text{EMP}_*'(i)(z(i)) \wedge \text{DEPT}'(i)(y)])$
$\rightarrow \lambda z \exists x[\text{DEPT}_*'(i)(x(i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x(i),\text{Shoes}) \wedge \text{AS-1}(z(i),x) \wedge \text{EMP}_*'(i)(z(i))]$
[it-0-NOM] #work-for a department such that it sells shoes ==>
$\exists x[\text{DEPT}_*'(i)(x(i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x(i),\text{Shoes}) \wedge \text{AS-1}(x_0(i),x) \wedge \text{EMP}_*'(i)(x_0\ (i))]$
who works-for a department such that it sells shoes ==>
$\lambda P \exists y[y(i) = u \wedge P(i)(y)]\ (\lambda i \lambda x_0\ \exists x[\text{DEPT}_*'(i)(x(i)) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x(i),\text{Shoes}) \wedge \text{AS-1}(x_0(i),x) \wedge \text{EMP}_*'(i)(x_0(i))])$
$\rightarrow \exists x[\text{EMP}_*'(i)(u) \wedge \text{DEPT}_*'(i)(x(i)) \wedge \text{AS-1}(u,x) \wedge \text{ITEM}_*'(i)(\text{Shoes}) \wedge \text{REL-2}(x(i),\text{Shoes})]$

Yes-No questions can take two forms in the fragment:

**(6-10) Is it the case that Peter earns 30K?**

and The two derivation trees are as follows:

**(6-11) Does Peter earn 30K?**

```
                Is it the case that Peter earns 30K?   S104
                                  |
                                  |
                Is it the case that Peter #earn 30K?   S101
                                  |
                                  |
                          Peter #earn 30K
```

and

```
                Does Peter earn 30K?    S104
                   |
                   |
                #AUX Peter earn 30K?    S101
                   |
                   |
                  Peter #earn 30K
```

Both of these questions are formed using one of the operations of rule **S101. [YNQ Formation]**

Peter #earn 30K → *(as above)* $\exists x[\text{AS-1}(\text{Peter},x) \wedge \text{EMP}_*'(i)(\text{Peter}) \wedge \text{SAL}'(i)(x) \wedge x(i) = 30\text{K}]$

Is it the case that Peter #earn 30K? ==> $\exists x[\text{AS-1}(\text{Peter},x) \wedge \text{EMP}_*'(i)(\text{Peter}) \wedge \text{SAL}'(i)(x) \wedge x(i) = 30\text{K}]$

Is it the case that Peter earns 30K? ==> $\exists x[\text{EMP}_*'(i)(\text{Peter}) \wedge \text{SAL}'(i)(x) \wedge x(i) = 30\text{K} \wedge \text{AS-1}(\text{Peter},x)]$

or alternatively:

Does Peter earn 30K? ==> $\exists x[\text{EMP}_*'(i)(\text{Peter}) \wedge \text{SAL}'(i)(x) \wedge x(i) = 30\text{K} \wedge \text{AS-1}(\text{Peter},x)]$

"When" questions are illustrated by the following example

**(6-12) When did Peter earn 25K?**

which makes use of rule **S110. [Tensed WHENQ Formation]**.

Here is the derivation and corresponding translation:

```
        When did Peter earn 25K?   S110
             /        \
            /          \
         when      Peter #earn 25K
```

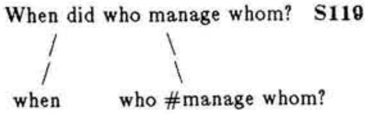
Peter #earn 25K → *(as above)* $\exists y[\text{AS-1}(\text{Peter},y) \wedge \text{EMP}_*'(i)(\text{Peter}) \wedge \text{SAL}'(i)(y) \wedge y(i) = 25\text{K}]$

When did Peter earn 25K? ==> $\lambda p \lambda i_1[[i_1 < i] \wedge p(i_1)](\lambda i \exists y[\ \text{AS-1}(\text{Peter},y) \wedge \text{EMP}_*'(i)(\text{Peter}) \wedge \text{SAL}'(i)(y) \wedge y(i) = 25\text{K}])$

→ $\lambda i_1 \exists y[[i_1 < i] \wedge \text{EMP}_*'(i_1)(\text{Peter}) \wedge \text{SAL}'(i_1)(y) \wedge y(i_1) = 25\text{K} \wedge \text{AS-1}(\text{Peter},y)]$

The next example illustrates the interaction of "when" and an already-formed Term question:

**(6-13) When did who manage whom?**



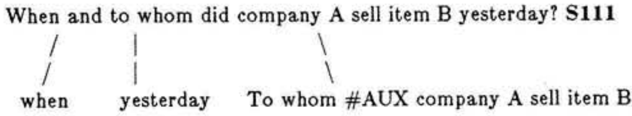
Who #manage whom? → $\exists y[\mathrm{EMP}_*'(i)(u_2) \wedge y(i) = u_1 \wedge \text{AS-1}(u_2,y) \wedge \mathrm{MGR}'(i)(y)]$

When did who manage whom? ==> $\lambda p \lambda i_1[[i_1 < i] \wedge p(i_1)]\ (\lambda i \exists y[\mathrm{EMP}_*'(i)(u_2) \wedge y(i) = u_1 \wedge \text{AS-1}(u_2,y) \wedge \mathrm{MGR}'(i)(y)])$

→ $\lambda i_1 \exists y[[i_1 < i] \wedge \mathrm{EMP}_*'(i_1)(u_2) \wedge \mathrm{MGR}'(i_1)(y) \wedge y(i_1) = u_1 \wedge \text{AS-1}(u_2,y)]$

Finally, the following example illustrates the introduction of "when" when the first question word is in the dative, and also how "when" questions interact with time phrases:

**(6-14) When and to whom did company A sell item B yesterday?**


When and to whom did company A sell item B yesterday? S111
when
yesterday
To whom #AUX company A sell item B?

To whom #AUX company A sell item B? → *(as above)*

$\exists x[\mathrm{DEPT}_*'(i)(u_1) \wedge x(i) = u_1 \wedge \mathrm{COMP}_*'(i)(A) \wedge \mathrm{ITEM}_*'(i)(B) \wedge \text{REL-3}(A,B,u_1)]$

When and to whom did company A sell item B yesterday? ==>

$\lambda p \lambda i_1[[i_1 < i] \wedge \mathrm{yesterday}'(i_1) \wedge p(i_1)](\lambda i \exists x[\mathrm{DEPT}_*'(i)(u_1) \wedge x(i) = u_1 \wedge \mathrm{COMP}_*'(i)(A) \wedge \mathrm{ITEM}_*'(i)(B) \wedge \text{REL-3}(A,B,u_1)])$

→ $\lambda i_1\ \exists x[[i_1 < i] \wedge \mathrm{yesterday}'(i_1) \wedge \mathrm{DEPT}_*'(i_1)(u_1) \wedge x(i_1) = u_1 \wedge \mathrm{COMP}_*'(i_1)(A) \wedge \mathrm{ITEM}_*'(i_1)(B) \wedge \text{REL-3}(A,B,u_1)]$

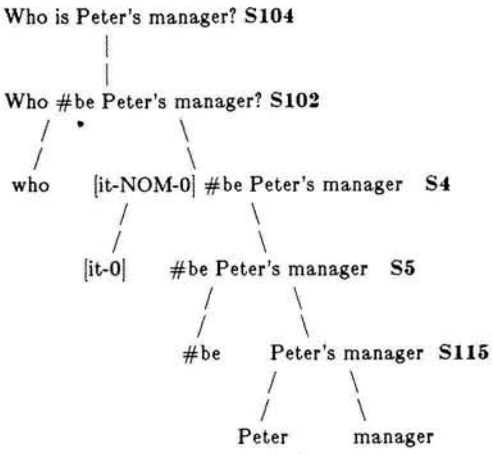
This example uses rule **S111.** [**Tensed WHENQ Formation with Specified Time**].

This concludes the examples of the kinds of queries expressible in the language QE-III, and the semantics and pragmatics that the fragment provides for them. We now present some of the other additions we have made to the PTQ fragment in order to express certain other common query constructions.

### 6.5. Miscellaneous Features of QE-III

The use of possessives is very common in database queries, and is easily incorporated into the fragment as the following rules and examples indicate:

**(6-15) Who is Peter's manager?**


Who is Peter's manager? S104
Who #be Peter's manager? S102
who
[it-NOM-0] #be Peter's manager S4
[it-0]
#be Peter's manager S5
#be
Peter's manager S115
Peter
manager

The phrase Peter's manager is formed by rule **S115. [Possessive Formation]**.

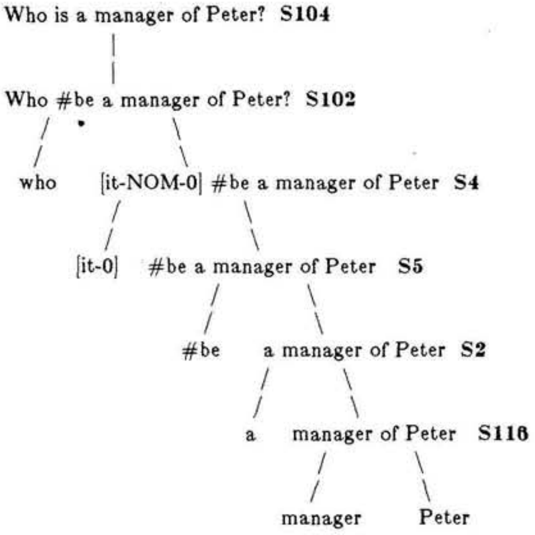
The resulting sentence is translated as follows:

Peter's ==> $\lambda W \lambda P \lambda Q \exists x[P(i)(x) \wedge Q(i)(x) \wedge W(i)[\lambda i \lambda y AS\text{-}1(y(i),x)]](\lambda i \lambda P \exists y[P(i)(y) \wedge y(i) = Peter])$
$\rightarrow \lambda P \lambda Q \exists x[P(i)(x) \wedge Q(i)(x) \wedge \exists y[AS\text{-}1(y(i),x) \wedge y(i) = Peter]]$
Peter's manager ==> $\lambda P \lambda Q \exists x[P(i)(x) \wedge Q(i)(x) \wedge \exists y[ AS\text{-}1(y(i),x) \wedge y(i) = Peter]](\lambda i\ MGR'(i))$
$\rightarrow \lambda Q \exists x[MGR'(i)(x) \wedge Q(i)(x) \wedge \exists y[AS\text{-}1(y(i),x) \wedge y(i) = Peter]]$
#be ==> $\lambda W \lambda z_1 W(i)(\lambda i \lambda z_2[z_1(i) = z_2(i)])$
#be Peter's manager ==> $\lambda W \lambda z_1 W(i)(\lambda i \lambda z_2[z_1(i) = z_2(i)])(\lambda i \lambda Q \exists x[MGR'(i)(x) \wedge Q(i)(x) \wedge \exists y[AS\text{-}1(y(i),x) \wedge y(i) = Peter]])$
$\rightarrow \lambda z_1 \exists x[MGR'(i)(x) \wedge [z_1(i) = x(i)]) \wedge \exists y[AS\text{-}1(y(i),x) \wedge y(i) = Peter]])$
[it-NOM-0] #be Peter's manager → *(as above)* $\exists x[MGR'(i)(x) \wedge [x_0(i) = x(i)]) \wedge \exists y[AS\text{-}1(y(i),x) \wedge y(i) = Peter]])$
$\rightarrow \exists x[MGR'(i)(x) \wedge [x_0(i) = x(i)] \wedge AS\text{-}1(Peter,x)]$
who ==> $\lambda P \exists y[y(i) = u \wedge P(i)(y)]$
Who is Peter's manager ==> $\lambda P \exists y[y(i) = u \wedge P(i)(y)](\lambda i \lambda x_0 \exists x[MGR'(i)(x) \wedge [x_0(i) = x(i)] \wedge AS\text{-}1(Peter,x)])$
$\rightarrow \exists x[MGR'(i)(x) \wedge x(i) = u \wedge AS\text{-}1(Peter,x)]$

An alternative way of phrasing the above question uses "of" instead of the possessive marker:

**(6-16) Who is a manager of Peter?**

This sentence makes use of Rule **S116. [Attributive Phrase Formation]**.


Who is a manager of Peter? S104
Who #be a manager of Peter? S102
who
[it-NOM-0] #be a manager of Peter S4
[it-0]
#be a manager of Peter S5
#be
a manager of Peter S2
a
manager of Peter S116
manager
Peter

and ultimately receives the same translation:

manager of Peter $==> \lambda x \lambda P \exists y[P(i)(y) \wedge y(i) = \text{Peter}](\lambda i \lambda z \text{MGR'}(i)(x) \wedge \text{AS-1}(z(i),x))$

$\quad \rightarrow \lambda x \exists y[\text{MGR'}(i)(x) \wedge \text{AS-1}(y(i),x) \wedge y(i) = \text{Peter}]$

a manager of Peter $\rightarrow \lambda Q \exists z[Q(i)(z) \wedge \exists y[\text{MGR'}(i)(z) \wedge \text{AS-1}(y(i),z) \wedge y(i) = \text{Peter}]]$

#be a manager of Peter $\rightarrow \lambda z_1 \lambda Q \exists z[Q(i)(z) \wedge \exists y[\text{MGR'}(i)(z) \wedge \text{AS-1}(y(i),z) \wedge y(i) = \text{Peter}]]) (\lambda i \lambda z_2[z_1(i) = z_2(i)])$

$\quad \rightarrow \lambda z_1 \exists z[[z_1(i) = z(i)] \wedge \exists y[\text{MGR'}(i)(z) \wedge \text{AS-1}(y(i),z) \wedge y(i) = \text{Peter}]]$

[it-NOM-0] #be a manager of Peter $\rightarrow \exists z[[x_0(i) = z(i)] \wedge \exists y[\text{MGR'}(i)(z) \wedge \text{AS-1}(y(i),z) \wedge y(i) = \text{Peter}]]$

Who is a manager of Peter $\rightarrow \exists x[x(i) = u \wedge \exists z[[x(i) = z(i)] \wedge \exists y[\text{MGR'}(i)(z) \wedge \text{AS-1}(y(i),z) \wedge y(i) = \text{Peter}]])$

$\quad \rightarrow \exists z[z(i) = u \wedge \text{MGR'}(i)(z) \wedge \text{AS-1}(\text{Peter},z)]$

Note that what might be considered an "equivalent"query in English,

**(6-17) Who manages Peter?**

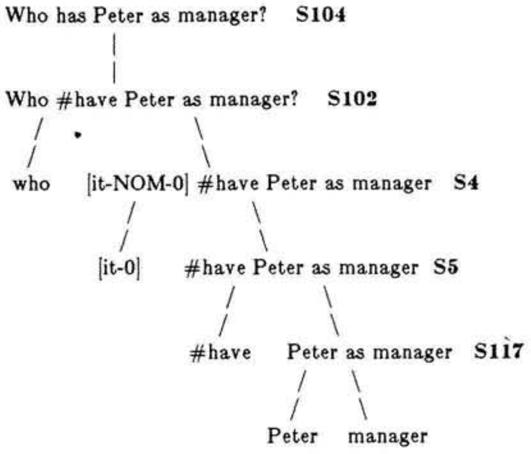
would be translated as:

$\exists z[\text{EMP}_*\text{'}(i)(\text{Peter}) \wedge \text{MGR'}(i)(z) \wedge z(i) = u \wedge \text{AS-1}(\text{Peter},z)]$

The difference between the translations of 6-15 and 6-16, on the one hand, and of 6-17, on the other, is that in these two examples no role (attribute) is specified for Peter. This is because the "neutral" verb "#be" is unable to specify the roles of its subject and object as the verb "#manage" can. This sort of situation could be rectified by means of a meaning postulate that, in this case, would specify the possible roles for an entity that "had" (was associated with) an IC that was a MGR (in this case only an EMPloyee can have a MGR.)

Roles can also be specified by means of the word "as":

**(6-18) Who has Peter as manager?**


Who has Peter as manager? S104
Who #have Peter as manager? S102
who
[it-NOM-0] #have Peter as manager S4
[it-0]
#have Peter as manager S5
#have
Peter as manager S117
Peter
manager

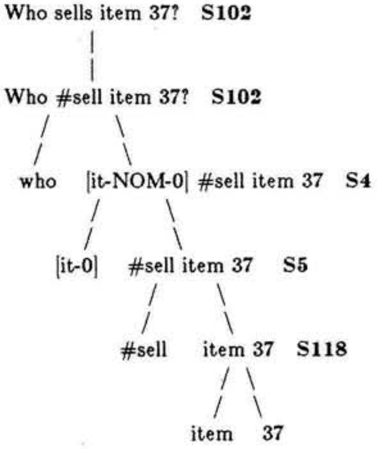
The rule for this kind of construction is **S117. [Role Specification for Term (I)]**.

This example also illustrates the general translation for the verb "#have."

Peter as manager ==> $\lambda Q(\lambda P\exists x[P(i)(x) \wedge x(i) = \mathrm{Peter}](\lambda i\lambda y Q(i)(y) \wedge \mathrm{MGR}'(i)(y)))$
$\quad \rightarrow \lambda Q\exists x[Q(i)(x) \wedge \mathrm{MGR}'(i)(x) \wedge x(i) = \mathrm{Peter}]$
#have ==> $\lambda W\lambda x[W(i)(\lambda i\lambda y[\mathrm{AS\text{-}1}(x(i),y)]]$
#have Peter as manager $\rightarrow \lambda x\lambda Q\exists z[Q(i)(z) \wedge \mathrm{MGR}'(i)(z) \wedge z(i) = \mathrm{Peter}](\lambda i\lambda y[\mathrm{AS\text{-}1}(x(i),y)])$
$\quad \rightarrow \lambda x\exists z[\mathrm{AS\text{-}1}(x(i),z) \wedge \mathrm{MGR}'(i)(z) \wedge z(i) = \mathrm{Peter}]$
[it-NOM-0] #have Peter as manager $\rightarrow \exists z[\mathrm{AS\text{-}1}(x_0(i),z) \wedge \mathrm{MGR}'(i)(z) \wedge z(i) = \mathrm{Peter}]$
Who has Peter as manager $\rightarrow \exists z[\mathrm{MGR}'(i)(z) \wedge z(i) = \mathrm{Peter} \wedge \mathrm{AS\text{-}1}(u,z)]$

A final form of role-specification in database queries takes the form of simple concatenation of the role and a Term, as illustrated in this example:

**(6-19) Who sells item 37?**



**S118. [Role Specification for Term (II)]** provides for this construction.

Here is the translation:

item ==> $\lambda W\lambda Q\ W(i)[\lambda i\lambda y[Q(i)(y) \wedge \mathrm{ITEM}_*'(i)(y(i))]]$

37 ==> $\lambda P \exists x[P(i)(x) \wedge x(i) = 37]$<br>
item 37 ==> $\lambda W \lambda Q\ W(i)[\lambda i \lambda y[Q(i)(y) \wedge \mathrm{ITEM}_*'(i)(y(i))]](\lambda i \lambda P \exists x[P(i)(x) \wedge x(i) = 37])$

→ $\lambda Q \exists x[Q(i)(x) \wedge \mathrm{ITEM}_*'(i)(x(i)) \wedge x(i) = 37]$

#sell ==> $\lambda W \lambda x[W(i)\ (\lambda i \lambda y_{\bullet}[\text{REL-2}(x(i),y(i)) \wedge \mathrm{DEPT}_*'(i)(x(i)) \wedge \mathrm{ITEM}_*'(i)(y(i))])]$

#sell item 37 → $\lambda x[\text{REL-2}(x(i),37) \wedge \mathrm{DEPT}_*'(i)(x(i)) \wedge \mathrm{ITEM}_*'(i)(37)]$

[it-NOM-0] #sell item 37 → $\text{REL-2}(x_0(i),37) \wedge \mathrm{DEPT}_*'(i)(x_0(i)) \wedge \mathrm{ITEM}_*'(i)(37)$

Who sells item 37? → *(as above)* $\mathrm{DEPT}_*'(i)(u) \wedge \mathrm{ITEM}_*'(i)(37) \wedge \text{REL-2}(u,37)$

## 7. Conclusion

The problem of modelling the semantics of time is one which is beginning to be explored by researchers in a number of different areas of Computer Science. We believe that formal logic can make an important contribution to our understanding and specification of the properties of time that we with to incorporate into our models and systems. Using the logic $IL_s$ and the framework of MS, we have presented in this paper an overview of the HRDM, which is a formalization of the concept of an historical database. HRDM provides for the storage of historical information, the specification of constraints on the way that information can change over time, and a query language for accessing that information with specific reference to its temporal dimension.

To augment the relational query language of HRDM, we have in this paper described a formal English database query language, QE-III, which is defined in a MS framework. QE-III incorporates an account of question semantics that accords with the semantics of HRDM, an account of the semantics of multiple-WH questions, an account of the semantics of time, and a grammar that is conducive to a computer implementation. In addition to its formal syntax and parallel semantics, QE-III is provided with a formal pragmatic component which provides a representation for the answer(s) to a question as a function of its syntax and semantics. We believe that this approach, and the whole area of **formal** pragmatics as a component of language theory, is a fertile area for further research.

## References

[Ariav et al. 84] Ariav, G., Beller, A., and Morgan, H.L.
*A Temporal Model.*
Technical Report DS-WP 82-12-05, Decision Sciences Dept., Univ. of Penn., December, 1984.

[Ben-Zvi 82] Ben-Zvi, J.
*The Time Relational Model.*
PhD thesis, Dept. of Computer Science, University of California, Los Angeles, 1982.
(Unpublished).

[Bennett 74] Bennett, M.R.
*Some Extensions of a Montague Fragment of English.*
PhD thesis, UCLA, 1974.
(Distributed by Indiana University Linguistic Club, Bloomington.).

[Bennett 79] Bennett, M.R.
*Questions in Montague Grammer.*
Technical Report, Indiana University Linguistics Club, Bloomington, 1979.

[Clifford 82a] Clifford, J.
A Model for Historical Databases.
In *Proceedings of Logical Bases for Data Bases.* ONERA-CERT, Toulouse, France, December, 1982.

[Clifford 82b] Clifford, J.
*A Logical Framework for the Temporal Semantics and Natural-Language Querying of Historical Databases.*
PhD thesis, Dept. of Computer Science, SUNY at Stony Brook, December, 1982.
(Unpublished).

[Clifford 85] Clifford, J.
Towards an Algebra of Historical Relational Databases.
In *ACM-SIGMOD International Conference on Management of Data.* ACM, Austin, May, 1985.

[Clifford 87] Clifford, J.
*Natural Language Querying of Historical Databases.*
Technical Report CRIS-#???, GBA-WP#???, Dept. of Information Systems, New York Univ., 1987.
(Submitted to **Computational Linguistics**).

[CliffordCroker 87]
Clifford, J., and Croker, A.
The Historical Relational Data Model (HRDM) and Algebra Based on Lifespans,
In *Proc. Third International Conference on Data Engineering.* IEEE, Los Angeles, February, 1987.

[CliffordWarren 83]
Clifford, J., and Warren D.S.
Formal Semantics for Time in Databases.
*ACM Trans. on Database Systems* 6(2):214-254, June, 1983.

[Codd 70] Codd, E.F.
A Relational Model of Data For Large Shared Data Banks.
*Comm. of the ACM* 13(6):377-387, June, 1970.

[Dowty 78] Dowty, D.R.
*A Guide to Montague's PTQ.*
Technical Report, Indiana University Linguistics Club, Bloomington, 1978.

[Dowty 79] Dowty, D.R.
*Word Meaning and Montague Grammer.*
D. Reidel Publishing Co., Dordrecht, 1979.

[Friedman 79] Friedman, J.
An Unlabelled Bracketing Solution to the Problem of Conjoined Phrases in Montaque's PTQ.
*Journal of Philosophical Logic* 8:151-169, 1979.

[GadiaVaishnav 85]
Gadia, S. K. and Vaishnav, J.
A Query Language for a Homogeneous Temporal Database.
In *Proc. of The Fourth Annual ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, pages 51-56. 1985.

[Gallin 75] Gallin, D.
*Intensional and Higher-Order Modal Logic.*
North-Holland, Amsterdam, 1975.

[Karttunen 77] Karttunen, L.
Syntax and Semantics of Questions.
*Linguistics and Philosophy* 1:3-44, 1977.

[Klopprogge 81] Klopprogge, M.R.
Term: An Approach to Include the Time Dimension in the Entity-Relationship Model.
In P.P.S. Chen (editor), *Entity-Relationship Approach to Information Modeling and Analysis*, pages 477-512. ER Institute, 1981.

[KlopproggeLockemann 83]
Klopprogge, M.R., and Lockemann, P.C.
Modelling Information Preserving Databases: Consequences of the Concept of Time.
In *Proc. of The Ninth International Conference on Very Large Data Bases*, pages 399-416. 1983.

[Lum et al. 84] Lum, V., et al.
Designing DBMS Support for the Temporal Dimension.
In *ACM-SIGMOD International Conference on Management of Data*, pages 115-126. Boston, June, 1984.

[Montague 70] Montague, R.
English as a Formal Language.
In B. Visentini et al (editors), *Linguaggi nella Societa e nella Tecnica*, pages 189-224. Edizioni di Comunita, Milan, 1970.
Reprinted in Montague 1974.

[Montague 73] Montague, R.
The Proper Treatment of Quantification in English.
*Approaches to Natural Language.*
D. Reidel Publishing Co., Dordrecht, 1973, pages 221-242.
Reprinted in Montague 1974.

[Partee 75] Partee, B.
Montague Grammar and Transformational Grammar.
*Linguistic Inguiry* 6(2):203-300, 1975.

[Snodgrass 84] Snodgrass, R.
The Temporal Query Language TQuel.
In *Proceedings of the 3rd ACM SIGACT-SIGMOD Symp. on Principles of Database Systems*, pages 204-212. Waterloo, Ontario, Canada, April, 1984.

[SnodgrassAhn 85]
Snodgrass, R. and Ahn, I.
A Taxonomy of Time in Databases.
In *ACM-SIGMOD International Conference on Management of Data*, pages 236-246. Austin, TX, May, 1985.

[Thomason 72a] Thomason, R.
*On The Semantic Interpretation of the Thomason 1972 Fragment.*
Technical Report, Indiana University Linguistics Club, Bloomington, 1972.

[Thomason 72b] Thomason, R.
Some Extensions of Montague Grammar.
*Montague Grammar.*
Academic Press, New York, 1972.