2.

Control/Query

Chapter Chairs/Editors: Mark Shafarman
Oacis Healthcare Systems, Inc.

Pat B. Cahill
Mayo Foundation

Mark Tucker
Regenstrief Institute

2.1 INTRODUCTION

The Control/Query chapter of this Standard defines the generic rules that apply to all messages. Subsequent sections define functionally specific messages to be exchanged among certain applications. The specific aspects of message definition that are addressed herein are:

a) the form to be used in functional chapters for describing messages. This includes their purpose, their contents, and the interrelationships among them. This form is called an **abstract message definition** because it is purely a level 7 (application) definition.

b) the HL7 encoding rules for converting an abstract message into a string of characters that comprises an actual message

c) the programming procedures required to exchange messages using the HL7 specifications

d) the anticipated relationship with lower level protocols

e) certain message segments that are components of all messages

f) a single message, the acknowledgment message, that may be used unchanged in multiple applications

2.2 CONCEPTUAL APPROACH

2.2.1 Trigger events

The Standard is written from the assumption that an event in the real world of healthcare creates the need for data to flow among systems. The real-world event is called the **trigger event**. For example, the trigger event **a patient is admitted** may cause the need for data about that patient to be sent to a number of other

systems. The trigger event, **an observation (e.g., a CBC result) for a patient is available,** may cause the need for that observation to be sent to a number of other systems. When the transfer of information is initiated by the application system that deals with the triggering event, the transaction is termed an **unsolicited update**.

Note:	No assumption is made about the design or architecture of the application system creating the unsolicited update. The scope of HL7 is restricted to the specification of messages between application systems, and the events triggering them.

HL7 allows the use of trigger events at several different levels of data granularity and inter-relationships. For example, most Patient Administration (ADT) trigger events concern single objects (such as an admit event, which creates a message that contains data about a single person and/or account). Other ADT trigger events are concerned with relationships between more than one object (e.g., the merge events, which specify patient or account merges). Some ADT trigger events pertain to a collection of objects that may have no significant inter-relationships (e.g., a record-oriented location-based query, whose response contains data about a collection of inpatients who are related only temporarily by local geography).

2.2.2 Acknowledgments: original mode

When the unsolicited update is sent from one system to another, this acknowledgment mode specifies that it be acknowledged at the application level. The reasoning is that it is not sufficient to know that the underlying communications system guaranteed delivery of the message. It is also necessary to know that the receiving application processed the data successfully at a logical application level.

The acknowledgment may contain data of interest to the system that initiated the exchange. For example, if a patient care system has processed the trigger event **a lab test is ordered for a patient**, it may send an unsolicited update to a lab application identifying the patient, the test ordered, and various other information about the order. The ancillary system will acknowledge the order when it has processed it successfully. For some pairings of patient care and ancillary department systems the acknowledgment may also include the ancillary identification number that was assigned. (HL7 does not require Order Entry and Results Reporting applications to interface in this manner, but it supports those that do.)

The HL7 Standard makes no assumptions about the ownership of data. It also makes no requirements of its own on the subsequent action of the recipient of data, nor does it make any assumption about the design or architecture of the receiving application system. The scope of HL7 is restricted to the specification of messages between application systems, and the events triggering them. HL7 does not explicitly support, but can be used with, systems that support store and forward and data broadcast facilities (see the HL7 Implementation Guide).

The HL7 Standard makes no functional interpretation of the requirement that a system commit the data in a message to its database before acknowledging it. All that is required is that the receiving system accept responsibility for the data, providing the same integrity test that it would apply to data from any source. To continue the prior example, the ancillary system may acknowledge the order after placing it in an input queue, expecting to fully process the order into its database at a future time. The only assumption is that the input queue is maintained at the same level of integrity as the database.

2.2.3 Acknowledgments: enhanced mode

The HL7 acknowledgment paradigm has been extended to distinguish both accept and application acknowledgments, as well the conditions under which each is required. With a positive accept acknowledgment, the receiving system commits the message to safe storage in a manner that releases the sending system from the need to resend the message. After the message has been processed by the

receiving system, an application acknowledgment may be used to return the resultant status to the sending system.

2.2.4 Queries

A different data exchange occurs when one system sends a query to another. For example, in a cardiac catheterization application, there may be a trigger event **a procedure is scheduled** for a patient who is not already registered in the cardiac catheterization application's database. The application may send a request message containing the patient's ID number to the Patient Administration (ADT) system and receive a response containing the necessary data to permit processing of the order. This requesting transaction is a **query**, as distinguished from the unsolicited update discussed above. The information that flows between the systems is contained in the response. The response itself is not acknowledged with a third message.

In all cases, the HL7 Standard consists of a simple exchange of messages between a pair of applications: the unsolicited update and its acknowledgment or the query and its response. The underlying operational model is that of a **client** and a **server**. An application interfaces with another application using an event code that identifies the transaction. The other application responds with a message that includes data or an error indication. The initiating application may receive a reject status from the other application or from lower level software indicating that its message was not received correctly.

HL7 queries can be formulated using one of several methods:

1. HL7 "query filters," defined via the QRD and QRF segments. These are supported as in previous releases of HL7, and are referred to as "original mode" queries.

2. Embedded Query Language select statements, which enable the querying system to format the request as a free-form[1] query statement, using the query language of choice (e.g., SQL).

3. Virtual Table Request, which is similar in function to the Embedded Query Language message, but more rigorously formatted with delimiters.

4. Stored Procedure Requests, which invoke units of program code on the responding system that are built to satisfy a specific query (e.g., predefined queries, SQL stored procedures).

   Since the predefined queries supported by HL7 are limited in number and precisely-defined, each has a corresponding stored procedure name and parameter list associated with it. Refer to the functional chapters for the lists of supported queries.

5. Event Replay Queries, which are requests for data formatted as event messages.

HL7 includes SQL select statements as an alternate means of encoding query selection criteria. This alternative is offered as a convenience to implementors, and in no way implies that server systems must support generic SQL or be based on relational database technology.

These queries are defined in the appropriate chapters of this specification.

---

1 Although referred to as "free-form," the functional chapters of this specification define this field for commonly used queries.

2.3 COMMUNICATIONS ENVIRONMENT

The HL7 Standard defines the messages as they are exchanged among applications entities and the procedures used to exchange them. As such, it conceptually operates at the seventh level of the ISO model for Open System Interconnection (OSI). It is primarily concerned with the data content and interrelationship of messages and with communicating certain application-level error conditions.

Since the OSI protocols are not universally implemented, the HL7 Working Group is interested in providing standards that will be useful in the interim. It is also recognized that there is now, and will continue to be, interest in communicating health data among systems operating in communications environments that provide a high level of functionality, but use protocols other than ISO OSI. The universe of environments of interest to HL7 includes, but is not restricted to:

a) ad hoc environments that do not provide even basic transport reliability. Such environments consist of point-to-point RS-232 links, modems, and even LANs, if their connection to host computers is made via RS-232 communications links. Until OSI high level standards become truly prevalent, many healthcare interfaces will be implemented over such links. In such an environment, the HL7 Lower Level Protocols (LLP) may be used between systems to enhance the capabilities of the communications environment. The HL7 Lower Level Protocols are defined in the HL7 Implementation Guide, which is not an official part of the Standard.

b) environments that support a robust transport level, but do not meet the high level requirements. This includes environments such as TCP/IP, DECNET, and SNA.

c) ISO and proprietary networks that implement up to presentation and other high level services. IBM's SNA LU6.2 and SUN Microsystems's NFS are examples of complete proprietary networks.

d) two or more applications running on the same physical and/or logical machine that are not tightly integrated. In these environments, the messaging capabilities may be provided by inter-process communications services (e.g., Pipes in a UNIX System).

The HL7 Standard assumes the communications environment will provide the following capabilities:

a) error free transmission. Applications can assume that they correctly received all of the transmitted bytes in the correct order that they were sent. This implies that error checking is done at a lower level. However, sending applications may not assume that the message was actually received without receiving an acknowledgment message.

b) character conversion. If the two machines exchanging data use different representations of the same character set, the communications environment will convert the data from one representation to the other.

c) message length. HL7 sets no limits on the maximum size of HL7 messages. The Standard assumes that the communications environment can transport messages of any length that might be necessary. In practice, sites may agree to place some upper bound on the size of messages and may use the message continuation protocol, described later in this chapter, for messages that exceed the upper limit.

> **Note:** Just as HL7 makes no assumptions about the design or architecture of the application systems sending and receiving HL7 messages, it makes no assumptions about the communications environment beyond those listed above. In particular, aside from the above assumptions, the communications environment, including its architecture, design and implementation, is outside the scope of HL7.

2.4 HL7 MESSAGES

This section and Sections 2.5, "SEGMENTS," through 2.9, "USE OF ESCAPE SEQUENCES IN TEXT FIELDS," define the components of messages and provide the methodology for defining abstract messages that are used in later chapters. A **message** is the atomic unit of data transferred between systems. It is comprised of a group of segments in a defined sequence. Each message has a **message type** that defines its purpose. For example the ADT Message type is used to transmit portions of a patient's Patient Administration (ADT) data from one system to another. A three-character code contained within each message identifies its type. These are listed in the Message Type list, Appendix A.

The real-world event that initiates an exchange of messages is called a trigger event. (See Section 2.2.1, "Trigger events," for a more detailed description of trigger events.) Appendix A contains the codes that represent all defined trigger events. These codes represent values such as **A patient is admitted** or **An order event occurred**. There is a one-to-many relationship between message types and trigger event codes. The same trigger event code may not be associated with more than one message type; however a message type may be associated with more than one trigger event.

All message types and trigger event codes beginning with the letter "Z" are reserved for locally-defined messages. No such codes will be defined within the HL7 Standard.

This section defines the components of messages and provides the methodology for defining abstract messages that is used in later chapters.

2.5 SEGMENTS

A **segment** is a logical grouping of **data fields**. Segments of a message may be required or optional. They may occur only once in a message or they may be allowed to repeat. Each segment is given a name. For example, the ADT message may contain the following segments: Message Header (MSH), Event Type (EVN), Patient ID (PID), and Patient Visit (PV1).

Each segment is identified by a unique three-character code known as the Segment ID. Although the actual segments are defined in various chapters, the ID codes assigned to all segments are listed in Appendix A.

All segment ID codes beginning with the letter **Z** are reserved for locally-defined messages. No such codes will be defined within the HL7 Standard.

2.6 FIELDS

A field is a string of characters. HL7 does not care how systems actually store data within an application. When fields are transmitted, they are sent as character strings. Except where noted, HL7 data fields may take on the null value. Sending the null value, which is transmitted as two double quote marks (""), is different from omitting an optional data field. The difference appears when the contents of a message will be used to update a record in a database rather than create a new one. If no value is sent, (i.e., it is omitted) the old value should remain unchanged. If the null value is sent, the old value should be changed to null. (For further details, see Section 2.10, "Message construction rules," - step 2d.)

The various chapters of the Standard contain segment attribute tables. These tables list and describe the data fields in the segment and characteristics of their usage. A comprehensive data dictionary of all HL7 fields is provided in Appendix A. In defining a segment, the following information is specified about each field:

2.6.1 Position (sequence within the segment)

Ordinal position of the data field within the segment. This number is used to refer to the data field in the text comments that follow the segment definition table. In the segment attribute tables this information is in a column labeled **SEQ**.

2.6.2 Maximum length

Maximum number of characters that one occurrence of the data field may occupy. The maximum length is not of conceptual importance in the abstract message or the HL7 coding rules. The length of a field is normative. However, in general practice it is often negotiated on a site-specific basis. It is calculated to include the component and subcomponent separators that are defined below. Because the maximum length is that of a single occurrence, the repetition separator is not included in calculating the maximum length (See Section 2.6.5, "Repetition"). In the segment attribute tables this information is in a column labeled **LEN.**

2.6.3 Data type

Restrictions on the contents of the data field. There are a number of data types defined by HL7. These are explained in Section 2.8, "DATA TYPES." In the segment attribute tables this information is in a column labeled **DT**.

2.6.4 Optionality

Whether the field is required, optional, or conditional in a segment. The designations are:

R - required

O - optional

C - conditional on the trigger event or on some other field(s). The field definitions following the segment attribute table should specify the algorithm that defines the conditionality for this field.

X - not used with this trigger event

B - left in for backward compatibility with previous versions of HL7. The field definitions following the segment attribute table should denote the optionality of the field for prior versions.

Note: For Versions 2.3 and higher: the optionality of fields should be explicitly documented in the segment field definitions that follow each segment definition table; if the optionality of fields within a segment changes depending on the trigger event, that optionality should also be explicitly documented.

Note: For fields defined by HL7 data types containing multiple components or subcomponents, the optionality of a given component or subcomponent must be specified in the detailed field definitions that follow the formal segment attribute tables. (See also Sections 2.7, "MESSAGE DELIMITERS," 2.8, "DATA TYPES," and 2.10, "MESSAGE CONSTRUCTION RULES").

In the segment attribute tables this information is in a column labeled **OPT**.

2.6.5 Repetition

Whether the field may repeat. The designations are:

N	-	no repetition
Y	-	the field may repeat an indefinite or site-determined number of times
(integer)	-	the field may repeat up to the number of times specified in the integer

Each occurrence may contain the number of characters specified by the field's maximum length. (See Section 2.6.2, "Maximum length.") In the segment attribute tables this information is in a column labeled **RP/#.**

2.6.6 Table

HL7 defines a table of values for this field. An entry in the table number column means that the table name and the element name are equivalent.

The manner in which HL7 defines the valid values for tables will vary. Certain fields, like Patient Location, will have values that vary from institution to institution. Such tables are designated user- or site-defined. Even though these tables are not defined in the Standard, they are given a user-defined table number to facilitate implementations. The IS data type is often used to encode values for these tables. Note that some of these tables (e.g., location) may reference common master files.

Others, like Event Type (*HL7 table 0003*), are a part of the HL7 Standard because they affect the interpretation of the messages that contain them. They are limited to the values established by the HL7 Standard. The ID data type is most often used to encode values for HL7 tables. When an HL7 table exists it is strongly recommended that it be used. The values are listed in Appendix A. These HL7 tables also appear in the text in a standard box format (e.g., the *HL7 table 0003 - Event type*). Additions may be included on a site-specific basis.

Still other tables contain values that are encoded by reference to other standards documents. For example, the encoding for Lab procedures is defined by ASTM E1238-94. The CE data type is used to encode values for these tables.

Finally, there are some user-defined tables that contain values that might be standardized across institutions but for which no applicable official standard exists. For these a set of **suggested** values may be listed in Appendix A. These suggested values appear in the text in a standard non-box format (e.g., *HL7 table 0062 - Event reason code* in Section 3.3.1.4, "Event reason code"). It is expected that these values will be used where applicable within an institution and serve as a basis for extensions as required. The appropriate functional committee within HL7 solicits suggestions for additional values from institutions that are applying the Standard.

Various HL7 data types (AD, CD, CE, CF, CK, CM, CN, CP, CQ, CX, DLN, ED, EI, FC, ID, IS, JCC, MO, HD, PL, PPN, PT, QSC, RI, RP, SCV, TQ, VH, XAD, XCN, XON, XPN, and XTN) are used to convey tabular values, or have a component containing tabular values. In the segment attribute tables this information is in a column labeled **TBL#.** The only exceptions are the CE and CF data types, which contain the table identifier as part of the data type definition.

2.6.7 ID number

Small integer that uniquely identifies the data field throughout the Standard. In the segment definition this information is in a column labeled **ITEM #.**

2.6.8 Name

Descriptive name for the field. In the segment attribute tables this information is in a column labeled **ELEMENT NAME**.

When the same name is used in more than one segment, it must have the same data type and semantic meaning in each segment, but it will have a distinct ID number (see Section 2.6.7, "ID number") in each distinct segment. To deal with any ambiguities arising from this convention, whenever a field is referenced herein, the segment name and position must always be included.

2.7 MESSAGE DELIMITERS

In constructing a message certain special characters are used. They are the segment terminator, the field separator, the component separator, subcomponent separator, repetition separator, and escape character. The segment terminator is always a carriage return (in ASCII, a hex 0D). The other delimiters are defined in the MSH segment, with the field delimiter in the 4th character position, and the other delimiters occurring as in the field called Encoding Characters, which is the first field after the segment ID. The delimiter values used in the MSH segment are the delimiter values used throughout the entire message. In the absence of other considerations, HL7 recommends the suggested values found in *Figure 2-1 delimiter values*.

At any given site, the subset of the possible delimiters may be limited by negotiations between applications. This implies that the receiving applications will use the agreed upon delimiters, as they appear in the Message Header segment (MSH), to parse the message.

Figure 2-1. Delimiter values

Delimiter	Suggested Value	Encoding Character Position	Usage
Segment Terminator	<cr> hex 0D	-	Terminates a segment record. This value cannot be changed by implementors.
Field Separator	|	-	Separates two adjacent data fields within a segment. It also separates the segment ID from the first data field in each segment.
Component Separator	^	1	Separates adjacent components of data fields where allowed.
Subcomponent Separator	&	4	Separates adjacent subcomponents of data fields where allowed. If there are no subcomponents, this character may be omitted.
Repetition Separator	~	2	Separates multiple occurrences of a field where allowed.
Escape Character	\	3	Escape character for use with any field represented by an ST, TX or FT data type, or for use with the data (fourth) component of the ED data type If no escape characters are used in a message, this character may be omitted. However, it must be present if subcomponents are used in the message.

2.8 DATA TYPES

The data types in this section are listed in alphabetical order.

Note: For data types which contain multiple components or subcomponents, the examples given in this section do not specify the optionality of the component or subcomponents. This must be specified in the field definitions that follow the formal segment attribute tables to a maximum length of 64K.

Except for the TS data type and the maximum or minimum lengths for several other data types (CE, PN, TX, FT), the field length of HL7 attributes is specified in the segment attribute tables, and any specific length of the components or subcomponents of those attributes must be specified in the field definitions that follow the formal segment attribute tables. In general, HL7 does not specify the lengths of components and/or subcomponents.

The data type examples in this Standard are given using the standard HL7 encoding rules, with the delimiter values from *Figure 2-1* of Section 2.7, "MESSAGE DELIMITERS." Although only one set of encoding rules is defined as a standard in HL7 Version 2.3, other encoding rules are possible (but since they are non-standard, they may only be used by a site-specific agreement).

In certain data type definitions, square brackets, "[" and "]", are used to specify optional parts of a data type (or of a data type component or subcomponent).

Figure 2-2. HL7 data types

Data Type Category/ Data type	Data Type Name	HL7 Section Reference	Notes/Format
Alphanumeric			
ST	String	2.8.38	
TX	Text data	2.8.43	

FT	Formatted text	2.8.17	
Numerical			
CQ	Composite quantity with units	2.8.9	<quantity (NM)> ^ <units (CE)>
MO	Money	2.8.23	<quantity (NM)> ^ <denomination (ID)>
NM	Numeric	2.8.25	
SI	Sequence ID	2.8.36	
SN	Structured numeric	2.8.37	<comparator> ^ <num1 (NM)> ^ <separator/suffix> ^ <num2 (NM)>
Identifier			
ID	Coded values for HL7 tables	2.8.19	
IS	Coded value for user-defined tables	2.8.20	
HD	Hierarchic designator	2.8.18	<namespace ID (IS)> ^ <universal ID (ST)> ^ <universal ID type (ID)> Used only as part of EI and other data types.
EI	Entity identifier	2.8.15	<entity identifier (ST)> ^ <namespace ID (IS)> ^ <universal ID (ST)> ^ <universal ID type (ID)>
RP	Reference pointer	2.8.34	<pointer (ST) > ^ < application ID (HD)> ^ <type of data (ID)> ^ <*subtype (ID)*>
PL	Person location	2.8.26	<point of care (IS)> ^ <room (IS)> ^ <bed (IS)> ^ <facility (HD)> ^ < location status (IS)> ^ <person location type (IS)> ^ <building (IS)> ^ <floor (IS)> ^ <location description (ST)>
PT	Processing type	2.8.29	<processing ID (ID)> ^ <processing mode (ID)>
Date/Time			
DT	Date	2.8.13	YYYY[MM[DD]]
TM	Time	2.8.39	HH[MM[SS[.S[S[S[S]]]]]][+/-ZZZZ]
TS	Time stamp	2.8.42	YYYY[MM[DD[HHMM[SS[.S[S[S[S]]]]]]]][+/-ZZZZ] ^ <degree of precision>
Code Values			
CE	Coded element	2.8.3	<identifier (ST)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ST)> ^ <alternate text (ST)> ^ <name of alternate coding system (ST)>
CF	Coded element with formatted values	2.8.4	<identifier (ID)> ^ <formatted text (FT)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^ <alternate formatted text (FT)> ^ <name of alternate coding system (ST)>
CK	Composite ID with check digit	2.8.5	<ID number (NM)> ^ <check digit (NM)> ^ <code identifying the check digit scheme employed (ID)> ^ < assigning authority (HD)>
CN	Composite ID number and name	2.8.7	<ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^ <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <source table (IS)> ^ <assigning authority (HD)>
CX	Extended composite ID with check digit	2.8.10	<ID (ST)> ^ <check digit (ST)> ^ <code identifying the check digit scheme employed (ID)> ^ < assigning authority (HD))> ^ <identifier type code (IS)> ^ < assigning facility (HD)
XCN	Extended composite ID number and name	2.8.46	In Version 2.3, use instead of the CN data type. <ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^ <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR)

			(ST)> ^ <degree (e.g., MD) (ST)> ^ <source table (IS)> ^ <assigning authority (HD)> ^ <name type code (ID)> ^ <identifier check digit (ST)> ^ <code identifying the check digit scheme employed (ID)> ^ <identifier type code (IS)> ^ <assigning facility (HD)>
Generic			
CM	Composite	2.8.6	No new CM's are allowed after HL7 Version 2.2. Hence there are no new CM's in Version 2.3.
Demographics			
AD	Address	2.8.1	<street address (ST)> ^ < other designation (ST)> ^ <city (ST)> ^ <state or province (ST)> ^ <zip or postal code (ST)> ^ <country (ID)> ^ <address type (ID)> ^ <other geographic designation (ST)>
PN	Person name	2.8.28	<family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^ <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)>
TN	Telephone number	2.8.40	[NN] [(999)]999-9999[X99999][B99999][C any text]
XAD	Extended address	2.8.45	In Version 2.3, replaces the AD data type. <street address (ST)> ^ <other designation (ST)> ^ <city (ST)> ^ <state or province (ST)> ^ <zip or postal code (ST)> ^ <country (ID)> ^ < address type (ID)> ^ <other geographic designation (ST)> ^ <county/parish code (IS)> ^ <census tract (IS)>
XPN	Extended person name	2.8.48	In Version 2.3, replaces the PN data type. <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^ <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <name type code (ID) >
XON	Extended composite name and ID number for organizations	2.8.47	<organization name (ST)> ^ <organization name type code (IS)> ^ <ID number (NM)> ^ <check digit (NM)> ^ <code identifying the check digit scheme employed (ID)> ^ <assigning authority (HD)> ^ <identifier type code (IS)> ^ <assigning facility ID (HD)>
XTN	Extended telecommunications number	2.8.49	In Version 2.3, replaces the TN data type. [NNN] [(999)]999-9999 [X99999] [B99999] [C any text] ^ <telecommunication use code (ID)> ^ <telecommunication equipment type (ID)> ^ <email address (ST)> ^ <country code (NM)> ^ <area/city code (NM)> ^ <phone number (NM)> ^ <extension (NM)> ^ <any text (ST)>
Specialty/Chapter Specific			
Waveform			
CD	Channel definition	2.8.2	For waveform data only, see Chapter 7, Section 7.15.3. <channel identifier (*)> ^ <channel number (NM)> & <channel name (ST)>> ^ <electrode names (*) > ^ <channel sensitivity/units (*) > ^ <calibration parameters (*)> ^ <sampling frequency (NM)> ^ <minimum/maximum data values (*)>
MA	Multiplexed array	2.8.22	For waveform data only, see Chapter 7, Section 7.15.2. <sample 1 from channel 1 (NM)> ^ <sample 1 from channel 2 (NM)> ^ <sample 1 from channel 3 (NM)> ...~<sample 2 from channel 1 (NM)> ^ <sample 2 from channel 2 (NM)> ^ <sample 2 from channel 3 (NM)> ...~
NA	Numeric array	2.8.24	For waveform data only, see Chapter 7, Section

			7.15.1. <value1 (NM)> ^ <value2 (NM)> ^ <value3 (NM)> ^ <value4 (NM)> ^ ...
ED	Encapsulated data	2.8.14	Supports ASCII MIME-encoding of binary data. <source application (HD) > ^ <main type of data (ID)> ^ <data subtype (ID)> ^ <encoding (ID)> ^ <data (ST)>
Price data			
CP	Composite price	2.8.8	In Version 2.3, replaces the MO data type. <price (MO)> ^ <price type (ID)> ^ <from value (NM)> ^ <to value (NM)> ^ <range units (CE)> ^ <range type (ID)>

Patient Administration/Financial Information			
FC	Financial class	2.8.16	<financial class (ID)> ^ <effective date (TS)>
Extended Queries			
QSC	Query selection criteria	2.8.31	<name of field (ST)> ^ <relational operator (ID)> ^ <value (ST)> ^ <relational conjunction (ID)>
QIP	Query input parameter list:	2.8.30	<field name (ST) > ^ <value1 (ST) & value2 (ST) & value3 (ST) ...>
RCD	Row column definition:	2.8.32	<HL7 item number (ST)> ^ <HL7 data type (ST)> ^ <maximum column width (NM)>
Master Files			
DLN	Driver's license number	2.8.11	<license number (ST)> ^ <issuing state, province, country (IS)> ^ <expiration date (DT)
JCC	Job code/class	2.8.21	<job code (IS)> ^ <job class (IS)>
VH	Visiting hours	2.8.44	<start day range (ID)> ^ <end day range (ID)> ^ <start hour range (TM)> ^ <end hour range (TM)>
Medical Records/Information Management			
PPN	Performing person time stamp:	2.8.27	<ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^ <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <source table (IS)> ^ <assigning authority (HD)> ^ <name type code(ID)> ^ <identifier check digit (ST)> ^ <code identifying the check digit scheme employed (ID)> ^ <identifier type code (IS)> ^ <assigning facility (HD)> ^ < date/time action performed (TS)>
Time Series:			
DR	Date/time range	2.8.12	Scheduling Chapter Only: <range start date/time (TS)> ^ <range end date/time (TS)>
RI	Repeat interval	2.8.33	Scheduling Chapter Only: <repeat pattern (IS)> ^ <explicit time interval (ST)>
SCV	Scheduling class value pair	2.8.35	Scheduling Chapter Only: <parameter class (IS)> ^ <parameter value (IS)>
TQ	Timing/quantity	2.8.41	For timing/quantity specifications for orders, see Chapter 4, Section 4.4. <quantity (CQ)> ^ <interval (*)> ^ <duration (*)> ^ <start date/time (TS)> ^ <end date/time (TS)> ^ <priority (ID)> ^ <condition (ST)> ^ <text (TX)> ^ <conjunction (ID)> ^ <order sequencing (*)>

* for subcomponents of these elements please refer to the definition in the text.

2.8.1 AD - address

```
Components:  <street address (ST)> ^ < other designation (ST)> ^ <city (ST)> ^ <state or province (ST)> ^ <zip
              or postal code (ST)> ^ <country (ID)> ^ <address type (ID)> ^ <other geographic designation
              (ST)>
```

Example:

```
|10 ASH LN^#3^LIMA^OH^48132|
```

2.8.1.1 Street address (ST)

2.8.1.2 The street or mailing address of a person or institution. Other designation (ST)

Second line of address. In general, it qualifies address. Examples: Suite 555 or Fourth Floor.

2.8.1.3 City (ST)

2.8.1.4 State or province (ST)

State or province should be represented by the official postal service codes for that country.

2.8.1.5 Zip or postal code (ST)

Zip or postal codes should be represented by the official codes for that country. In the US, the zip code takes the form 99999[-9999], while the Canadian postal code takes the form A9A-9A9.

2.8.1.6 Country (ID)

Defines the country of the address. ISO 3166 provides a list of country codes that may be used[2].

2.8.1.7 Address type (ID)

Type is optional and defined by *HL7 table 0190 - Address type*.

2.8.1.8 Other geographic designation (ST)

Other geographic designation includes county, bioregion, SMSA, etc.

2 Available from ISO 1 Rue de Varembe, Case Postale 56, CH 1211, Geneva, Switzerland

Table 0190 - Address type

Value	Description
C	Current Or Temporary
P	Permanent
M	Mailing
B	Firm/Business
O	Office
H	Home
N	Birth (nee)
F	Country Of Origin

2.8.2 CD - channel definition

Components: <channel identifier *(<channel number (NM)> & <channel name (ST)>)* > ^ <electrode names (CM)> ^ <channel sensitivity/units (CM)> ^ <calibration parameters (CM)> ^ <sampling frequency (NM)> ^ <minimum/maximum data values (CM)>

This data type is used for labeling of digital waveform data. See Chapter 7, Section 7.15.3, "CD - channel definition," for a complete description of this data type.

2.8.3 CE - coded element

Components: <identifier (ST)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ST)> ^ <alternate text (ST)> ^ <name of alternate coding system (ST)>

Example:

```
|F-11380^CREATININE^I9^2148-5^CREATININE^LN|
```

This data type transmits codes and the text associated with the code. To allow all six components of a CE data type to be valued, the maximum length of this data type must be at least 60 (see Section 2.6.2, "Maximum length")**.**

2.8.3.1 Identifier (ST)

Sequence of characters (the code) that uniquely identifies the item being referenced by the <text>. Different coding schemes will have different elements here.

2.8.3.2 Text (ST)

Name or description of the item in question. E.g., myocardial infarction or X-ray impression. Its data type is string (ST).

2.8.3.3 Name of coding system (ST)

Each coding system is assigned a unique identifier. This component will serve to identify the coding scheme being used in the identifier component. The combination of the **identifier** and **name of coding system** components will be a unique code for a data item Each system has a unique identifier. ASTM E1238-94, Diagnostic, procedure, observation, drug ID, and health outcomes coding systems are identified in the tables in Section 7.1.4, "Coding schemes." Others may be added as needed. When an HL7 table is used for a CE data type, the ***name of coding system*** component is defined as ***HL7nnnn*** where ***nnnn*** is the HL7 table number.

2.8.3.4 Alternate components

These three components are defined analogously to the above for the alternate or local coding system. If the Alternate Text component is absent, and the Alternate Identifier is present, the Alternate Text will be taken to be the same as the Text component. If the Alternate Coding System component is absent, it will be taken to mean the locally-defined system.

Note:	The presence of two sets of equivalent codes in this data type is semantically different from a repetition of a CE-type field. With repetition, several distinct codes (with distinct meanings) may be transmitted.

Note:	For HL7-defined tables which have not been adopted from some existing standard, the third component, "name of coding system," is constructed by appending the table number to the string "HL7." Thus, the field *RXR-2-site*, is a CE data type which refers to HL7 table number 0163. Its "name of coding system" component is "HL70163".

Figures 7-2 and *7-3* list many diagnostic, procedure, observation, drug, and health outcomes coding systems. Guidelines for their use are presented in Chapter 7, Section 7.1, "Introduction and Overview."

2.8.4 CF - coded element with formatted values

This data type transmits codes and the formatted text associated with the code. This data type can be used to transmit for the first time the formatted text for the **canned text** portion of a report, for example, a standard radiologic description for a normal chest X-ray. The receiving system can store this information and in subsequent messages only the identifier need be sent. Another potential use of this data type is transmitting master file records that contain formatted text. This data type has six components as follows:

```
Components:  <identifier (ID)> ^ <formatted text (FT)> ^ <name of coding system (ST)> ^ <alternate identifier
              (ID)> ^ <alternate formatted text (FT)> ^ <name of alternate coding system (ST)>
```

The components, primary and alternate, are defined exactly as in the CE data type with the exception of the second and fifth components which are of the formatted text data type. Example:

```
OBX||CF|71020^CXR^CPMC||79989^

\H\Description:\N\\.sp\\ti+4\Heart is not enlarged.  There is no evi-
dence of

pneumonia, effusion, pneumothorax or any masses.\.sp+3\\H\Impression:

\N\\.sp\\.ti+4\Negative chest.^CPMC
```

2.8.5 CK - composite ID with check digit

```
Components:  <ID number (NM)> ^ <check digit (NM)> ^ <code identifying the check digit scheme employed (ID)> ^
              < assigning authority (HD)>
```

Example:

```
|128952^6^M11^ADT01|
```

This data type is used for certain fields that commonly contain check digits, e.g., *PID-3-patient ID (internal)*. If a site is not using check digits for a particular CK field, the second and third components are not valued.

2.8.5.1 ID number (NM)

2.8.5.2 Check digit (NM)

The check digit in this data type is not an add-on produced by the message processor. It is the check digit that is part of the identifying number used in the sending application. If the sending application does not include a self-generated check digit in the identifying number, this component should be valued null.

2.8.5.3 Code identifying the check digit scheme employed (ID)

The check digit scheme codes are defined in *HL7 table 0061 - Check digit scheme*.

Table 0061 - Check digit scheme

Value	Description
M10	Mod 10 algorithm
M11	Mod 11 algorithm

The algorithm for calculating a Mod10 check digit is as follows:

Assume you have an identifier = 12345. Take the odd digit positions, counting from the right, i.e., 531, multiply this number by 2 to get 1062. Take the even digit positions, starting from the right (i.e., 42), prepend these to the 1062 to get 421062. Add all of these six digits together to get 15. Subtract this number from the next highest multiple of 10, i.e., 20 - 15 to get 5. The Mod10 check digit is 5. The Mod10 check digit for 401 is 0; for 9999, it's 4; for 99999999, it's 8.

The algorithm for calculating a Mod11 check digit is as follows:

Terms

d = digit of number starting from units digit, followed by 10's position, followed by 100's position, etc.

w = weight of digit position starting with the units position, followed by 10's position, followed by 100's position etc. Values for w = 2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 6, 7, etc. (repeats for each group of 6 digits)

c = check digit

Calculation

(Step 1) m = sum of (d * w) for positions 1, 2, etc. starting with units digit

for d = digit value starting with units position to highest order

for w = weight value from 2 to 7 for every six positions starting with units digit

(Step 2) c1 = m mod 11

(Step 3) if c1 = 0 then reset c1 = 1

(Step 4) = (11 - c1) mod 10

Example:

```
if the number is 1234567, then the mod 11 check digit = 4
```

The calculations are:

```
M  = (7*2)+(6*3)+(5*4)+(4*5)+(3*6)+(2*7)+(1*2)
   = 14 + 18 + 20 + 20 + 18 + 14 + 2
   = 106
c1 = 106 mod 11
   = 7
c  = (11-c1) mod 10
   =  4 mod 10
   =  4
```

Other variants of these check digit algorithms exist and may be used by local bilateral site agreement.

2.8.5.4 Assigning authority (HD)

The assigning authority is a unique name of the system that creates the data. It is an HD data type. It is equivalent to the application ID of the placer or filler order number (see Chapter 4). Assigning authorities are unique across a given HL7 implementation.

2.8.6 CM - composite

A field that is a combination of other meaningful data fields. Each portion is called a **component**. The specific components of CM fields are defined within the field descriptions. Certain other composites have been separately identified and are described below. ***The CM data type is maintained strictly for backward compatibility and may not be used for the definition of new fields.*** Wherever a component of an HL7 field is itself an HL7 data type which contains components, its delimiters are demoted by one. Thus a component designated as a CE data type should be encoded as `<identifier & text & name of coding system>` (see Section 2.8.3, "CE - coded element"). Note that since HL7 delimiters are not recursive, an HL7 data type containing components cannot be a subcomponent. When this level of detail is needed, each component of the HL7 data type can be encoded as a separate subcomponent. For an example of this, see the encoding of the filler order number in the order sequencing component of the Timing/Quantity data type.

2.8.7 CN - composite ID number and name

```
Components:  <ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^
             <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <source
             table (IS)> ^ <assigning authority (HD)>
```

A field identifying a person both as a coded value and with a text name. For specific fields, individual sites may elect to omit the ID or the name. Example:

```
|12372^RIGGINS^JOHN^""^""^""^MD^ADT1|
|12372^^^^^^^ADT1|
|^RIGGINS^JOHN^""^""^""^MD|
```

2.8.7.1 ID number (ST)

Coded ID according to a user-defined table, defined by the 8th component. If the first component is present, either the source table or the assigning authority must be valued.

2.8.7.2 Family name (ST)

2.8.7.3 Given name (ST)

2.8.7.4 Middle initial or name (ST)

2.8.7.5 Suffix (ST)

Used to specify a name suffix (e.g., Jr. or III).

2.8.7.6 Prefix (ST)

Used to specify a name prefix (e.g., Dr.).

2.8.7.7 Degree (ST)

Used to specify an educational degree (e.g., MD).

2.8.7.8 Source table (IS)

Refer to *user-defined table 0297 - CN ID source* for suggested values. Used to delineate the first component.

2.8.7.9 Assigning authority (HD)

In this version an optional 9th component, the assigning authority (HD), has been added. It is an HD data type (see Section 2.8.18, "HD - hierarchic designator").

2.8.8 CP - composite price

```
Components:  <price (MO)> ^ <price type (ID)> ^ <from value (NM)> ^ <to value (NM)> ^ <range units (CE)> ^
              <range type (ID)>
```

Note	This data type is often used to define a repeating field within a given segment.

Example:

```
|100.00&USD^UP^0^9^min^P~50.00&USD^UP^10^59^min^P~10.00&USD^UP^60^999^P
~50.00&USD^AP~200.00&USD^PF

~80.00&USD^DC|
```

2.8.8.1 Price (MO)

The only required component; usually containing a decimal point. Note that each component of the MO data type (Section 2.8.23, "MO - money") is a subcomponent here.

```
Subcomponents of price:  <quantity> & <denomination>
```

2.8.8.2 Price type (ID)

A coded value, data type ID. Refer to *HL7 table 0205 - Price type* for valid values.

Table 0205 - Price type

Value	Description
AP	administrative price or handling fee
PF	professional fee for performing provider
UP	unit price, may be based on length of procedure or service
TF	technology fee for use of equipment
DC	direct unit cost
IC	indirect unit cost
TP	total price

2.8.8.3 From value (NM)

Each is an NM data type; together they specify the "range." The range can be defined as either time or quantity. For example, the range can indicate that the first 10 minutes of the procedure has one price. Another repetition of the data type can use the range to specify that the following 10 to 60 minutes of the procedure is charged at another price per; a final repetition can specify that the final 60 to N minutes of the procedure at a third price.

Note that, if the <`price type`> component is TP, both <`from value`> and <`to value`> may be null.

2.8.8.4 To value (NM)

See *From value* above.

2.8.8.5 Range units (CE)

A coded value, data type CE, defined by the standard table of units for either time or quantity (see for example, the tables in Section 7.1.4, "Coding schemes"). This describes the units associated with the range, e.g., seconds, minutes, hours, days, quantity (e.g., count); it is required if <`from value`> and <`to value`> are present.

```
Subcomponents of range units:  <identifier (ID)> & <text (ST)> & <name of coding system (ST)> & <alternate
             identifier (ID)> & <alternate text (ST)> & <name of alternate coding system (ST)>
```

2.8.8.6 Range type (ID)

Refers to HL7 table 0298 - CP range type for valid values.

Table 0298 - CP range type

Value	Description
P	Pro-rate. Apply this price to this interval, pro-rated by whatever portion of the interval has occurred/been consumed
F	Flat-rate. Apply the entire price to this interval, do not pro-rate the price if the full interval has not occurred/been consumed

2.8.9 CQ - composite quantity with units

```
Components:  <quantity (NM)> ^ <units (CE)>
```

In future versions, CQ fields should be avoided because the same data can usually be sent as two separate fields, one with the value and one with the units as a CE data type.

Examples:

```
|123.7^kg|       kilograms is an ISO unit

|150^lb&&ANSI+|  weight in pounds is a customary US unit defined within
ANSI+.
```

2.8.9.1 Quantity (NM)

2.8.9.2 Units (CE)

The units in which the quantity is expressed. Field-by-field, default units may be defined within the specifications. When the observation is measured in the default units, the units need not be transmitted. If the measure is recorded in units different from the default, the measurement units must be transmitted as the second component. If the units are ISO+ units, then units should be recorded as lowercase abbreviations as specified in Chapter 7. If the units are ANSI or local, the units and the source table must be recorded as specified in Chapter 7. But in these cases the component separator should be replaced by the subcomponent delimiter

```
Subcomponents for units: <identifier (ID)> & <text (ST)> & <name of coding system (ST)> & <alternate
             identifier (ID)> & <alternate text (ST)> & <name of alternate coding system (ST)>
```

2.8.10 CX - extended composite ID with check digit

```
Components: <ID (ST)> ^ <check digit (ST)> ^ <code identifying the check digit scheme employed (ID)> ^ <
             assigning authority (HD)  )> ^ <identifier type code (IS)> ^ < assigning facility (HD)
```

Example:

```
|1234567^4^M11^ADT01^MR^University Hospital|
```

2.8.10.1 ID (ST)

Defined as in the CK data type (see Section 2.8.5, “CK - composite ID with check digit”) except that an ST data type is allowed instead of an NM data type.

2.8.10.2 Check digit (ST)

Defined as in the CK data type (see Section 2.8.5, “CK - composite ID with check digit”) except that an ST data type is allowed instead of an NM data type. The check digit in this data type is not an add-on produced by the message processor. It is the check digit that is part of the identifying number used in the

sending application. If the sending application does not include a self-generated check digit in the identifying number, this component should be valued null.

2.8.10.3 Code identifying the check digit scheme employed (ID)

Defined as in the CK data type (see Section 2.8.5, "CK - composite ID with check digit"). Refer to *HL7 table 0061- Check digit scheme* for valid values.

Note: The check digit and code identifying check digit scheme are null if ID is alphanumeric.

2.8.10.4 Assigning authority (HD)

The assigning authority is a unique name of the system that creates the data. It is an HD data type. It is equivalent to the application ID of the placer or filler order number (see Chapter 4). Assigning authorities are unique across a given HL7 implementation.

2.8.10.5 Identifier type code (IS)

A code corresponding to the type of identifier. In some cases, this code may be used as a qualifier to the "Assigning authority" component. Refer to *user-defined table 0203 - Identifier type* for suggested values.

User-defined table 0203 - Identifier type

Value	Description
AM	American Express
AN	Account Number
BR	Birth Registry Number
DI	Diner's Club Card
DL	Driver's License Number
DN	Doctor Number
DS	Discover Card
EI	Employee Number
EN	Employer Number
GI	Guarantor Internal Identifier
GN	Guarantor External Identifier
MS	MasterCard
MA	Medicaid Number
MC	Medicare Number
MR	Medical Record Number
PI	Patient Internal Identifier
PN	Person Number
PT	Patient External Identifier
RR	Railroad Retirement Number

SS	Social Security Number
UPIN	Medicare/HCFA's Universal Physician Identification Numbers
VS	VISA
VN	Visit Number
XX	Organization Identifier

2.8.10.6 Assigning facility (HD)

```
Subcomponents: <namespace ID (IS)> & < universal ID (ST)> & <universal ID type (ID)>
```

Definition: The place or location identifier where the identifier was first assigned to the patient. This component is not an inherent part of the identifier but rather part of the history of the identifier: as part of this data type, its existence is a convenience for certain intercommunicating systems.

2.8.11 DLN - driver's license number

```
Components:  <license number (ST)> ^ <issuing state, province, country (IS)> ^ <expiration date (DT)
```

Definition: This field contains the driver's license information. For state or province refer to official postal codes for that country; for country refer to ISO 3166 for codes.

2.8.11.1 Driver's license number (as ST data type)

This field contains the driver's license number.

2.8.11.2 Issuing state, province, country (IS)

Issuing authority for driver's license. For state or province refer to official postal codes for that country; for country refer to ISO 3166 for codes. Refer to *user-defined table 0333 - Driver's license issuing authority*.

2.8.11.3 Expiration date (DT)

Expiration date (DT) for driver's license.

2.8.12 DR - date/time range

```
Components:  <range start date/time (TS)> ^ <range end date/time (TS)>
```

2.8.12.1 Range start date/time (TS)

Definition: The first component contains the earliest date/time (time stamp) in the specified range.

2.8.12.2 Range end date/time (TS)

The second component contains the latest date/time in the specified range. Note that the TS (time stamp) data type allows the specification of precision.

2.8.13 DT - date

```
Format: YYYY[MM[DD]]
```

In prior versions of HL7, this data type was always specified to be in the format YYYYMMDD. In the current and future versions, the precision of a date may be expressed by limiting the number of digits used with the format specification YYYY[MM[DD]]. Thus, YYYY is used to specify a precision of "year," YYYYMM specifies a precision of "month," and YYYYMMDD specifies a precision of "day."

By site-specific agreement, YYYYMMDD may be used where backward compatibility must be maintained.

Examples:

```
|19880704|

|199503|
```

2.8.14 ED - encapsulated data

```
Components:  <source application (HD) > ^ <type of data (ID)> ^ <data subtype (ID))> ^ <encoding (ID)> ^ <data
             (ST)>
```

This data type transmits encapsulated data from a source system to a destination system. It contains the identity of the source system, the type of data, the encoding method of the data, and the data itself. This data type is similar to the RP (reference pointer) data type of Section 2.8.34, "RP - reference pointer," except that instead of pointing to the data on another system, it contains the data which is to be sent to that system.

2.8.14.1 Source application (HD)

A unique name that identifies the system which was the source of the data. Identical format and restrictions as in reference pointer (see Section 2.8.34.2, "Application ID (HD)").

```
Subcomponents: <namespace ID (IS)> & < universal ID (ST)> & <universal ID type (ID)>
```

2.8.14.2 Type of data

Identical to "type of data" component in the reference pointer (RP) data type. (See Section 2.8.34.3, "Type of data (CM)").

Refer to *HL7 table 0191 - Type of referenced data* for valid values.

2.8.14.3 Subtype

Refer to *HL7 table 0291 - Subtype of referenced data* for valid values.

2.8.14.4 Encoding (ID)

The type of encoding, if present, used to represent successive octets of binary data as displayable ASCII characters. Refer to *HL7 table 0299 - Encoding* for valid values.

Table 0299 - Encoding

Value	Description
A	no encoding - data are displayable ASCII characters.
Hex	hexadecimal encoding - consecutive pairs of hexadecimal digits represent consecutive single octets.
Base64	encoding as defined by MIME (Multipurpose Internet Mail Extensions) standard RFC 1521. Four consecutive ASCII characters represent three consecutive octets of binary data. Base64 utilizes a 65-character subset of US-ASCII, consisting of both the upper and lower case alphabetic characters, digits "0" through "9," "+," "/," and "=.".

Base64 is defined as follows (adapted from MIME Internet standard RFC 1521, which has precedence over this description). Proceeding from left to right across a 24-bit input group (three octets), each 6-bit group is used as an index into an array of 64 printable characters. The character referenced by the index is placed in the encoded string. These characters are shown in *HL7 table 0290 - MIME base64 encoding characters*, and are selected so as to be universally representable.

Special processing is performed if fewer than 24 bits are available in an input group at the end of data. A full encoding quantum is always completed at the end of data. When fewer than 24 input bits are available in an input group, zero bits are added (on the right) to form an integral number of 6-bit groups.

Output character positions which are not required to represent actual input data are set to the character "=". Since all canonically encoded output is an integral number of octets, only the following cases can arise: (1) the final quantum of input is an integral multiple of 24 bits; here, the final unit of encoded output will be an integral multiple of 4 characters with no "=" padding, (2) the final quantum of input is exactly 8 bits; here, the final unit of encoded output will be two characters followed by two "="padding characters, or (3) the final quantum of input is exactly 16 bits; here, the final unit of encoded output will be three characters followed by one "=" padding character.

Table 0290 - MIME base64 encoding characters

Value/Code	Value/Code	Value/Code	Value/Code
0 A	17 R	34 I	51 z
1 B	18 S	35 j	52 0
2 C	19 T	36 k	53 1
3 D	20 U	37 l	54 2
4 E	21 V	38 m	55 3
5 F	22 W	39 n	56 4
6 G	23 X	40 o	57 5
7 H	24 Y	41 p	58 6
8 I	25 Z	42 q	59 7
9 J	26 a	43 r	60 8
10 K	27 b	44 s	61 9
11 L	28 c	45 t	62 +
12 M	29 d	46 u	63 /
13 N	30 e	47 v	
14 O	31 f	48 w	(pad) =
15 P	32 g	49 x	
16 Q	33 h	50 y	

The interpretation of the encoded octets by any of the encoding methods, beyond what is either implicit or specified in the represented data type (such as their ordering within 16-bit or 32-bit binary words on the

destination application), is determined by the destination application and is beyond the scope of this Standard.

2.8.14.5 Data (ST)

Displayable ASCII characters which constitute the data to be sent from source application to destination application. The characters are limited to the legal characters of the ST data type, as defined in Section 2.8.38, "ST - string data," and, if encoded binary, are encoded according to the method of Section 2.8.14.2, "Type of data."

If the encoding component (see Section 2.8.14.4, "Encoding (ID)") = 'A' (none), then the data component must be scanned before transmission for HL7 delimiter characters, and any found must be escaped by using the HL7 escape sequences defined in Section 2.9, "USE OF ESCAPE SEQUENCES IN TEXT FIELDS ." On the receiving application, the data field must be de-escaped after being parsed.

If the encoding component (see Section 2.8.14.4, "Encoding (ID)") does not equal 'A,' then, after encoding, the (encoded) data must be scanned for HL7 delimiter characters, and any found must be escaped by using the HL7 escape sequences. Only then can the component be added to the HL7 segment/message. On the receiving application, the data field must be de-escaped after being parsed out of the message before being decoded. This can be expressed as 'encode', 'escape', parse, 'de-escape', 'decode'.

2.8.15 EI - entity identifier

```
Components: <entity identifier (ST)> ^ <namespace ID (IS)> ^ <universal ID (ST)> ^ < universal ID type (ID)>
```

The entity identifier defines a given entity within a specified series of identifiers.

The specified series, the *assigning authority*, is defined by components 2 through 4. The assigning authority is of the hierarchic designator data type, but it is defined as three separate components in the EI data type, rather than as a single component as would normally be the case. This is in order to maintain backward compatibility with the EI's use as a component in several existing data fields. Otherwise, the components 2 through 4 are as defined in Section 2.8.18, "HD - hierarchic designator." Hierarchic designators are unique across a given HL7 implementation.

2.8.15.1 Entity identifier (ST)

The first component, *entity identifier*, is usually defined to be unique within the series of identifiers created by the *assigning authority*, defined by a hierarchic designator, represented by components 2 through 4. (See Section 2.8.18, "HD - hierarchic designator".)

2.8.15.2 Namespace ID (IS)

Refer to *user-defined table 0300 - Namespace ID* for suggested values.

2.8.15.3 Universal ID (ST)

The HD's second component, *universal ID* (UID), is a string formatted according to the scheme defined by the third component, *universal ID type* (UID type). The UID is intended to be unique over time within the UID type. It is rigorously defined. Each UID must belong to one of the specifically enumerated schemes for constructing UID's (defined by the UID type). The UID (second component) must follow the syntactic rules of the particular universal identifier scheme (defined by the third component).

2.8.15.4 Universal ID type (ID)

Refer to *HL7 table 0301 - Universal ID type* for valid values. See Section 2.8.18, "HD - hierarchic designator," for definition.

2.8.16 FC - financial class

```
Components:  <financial class (IS)> ^ <effective date (TS)>
```

2.8.16.1 Financial class (IS)

This component contains the financial class assigned to a person. Refer to *user-defined table 0064 - Financial class* for suggested values.

2.8.16.2 Effective date (TS)

This component contains the effective date/time of the person's assignment to the financial class specified in the first component.

2.8.17 FT - formatted text data

This data type is derived from the string data type by allowing the addition of embedded formatting instructions. These instructions are limited to those that are intrinsic and independent of the circumstances under which the field is being used. The actual instructions and their representation are described later in this chapter. ***The FT field is of arbitrary length (up to 64k)*** and may contain formatting commands enclosed in escape characters. Example:

```
|\.sp\(skip one vertical line)|
```

For additional examples of formatting commands see Section 2.9, "Use of escape sequences in text fields."

2.8.18 HD - hierarchic designator

```
Components:  <namespace ID (IS)> ^ <universal ID  (ST)> ^ <universal ID type (ID)>
```

The HD is designed to be a more powerful application identifier. It is also designed to be used either as a local version of a site-defined application identifier or a publicly-assigned UID. Syntactically, the HD is a group of two application identifiers: one defined by the first component, and one defined by the second and third components.

The HD allows any site to act as an assigning authority (on a local or user-defined basis), even if it technically does not have the right to issue new IDs within an identification scheme. HDs which have defined third components (defined UID types) must be unique within the series of ID's defined by that component.

Note: The HD is used in fields that in earlier versions of HL7 used the IS data type. Thus, a single component HD (only the first component valued) will look like a simple IS data type for older systems expecting a single component in the place of the HD data type.

If the first component for the HD data type is present, the second and third components are optional. If the third component is present, then the second must also be present (although in this case the first is optional). The second and third components must either both be valued (both non-null), or both be not valued (both null).

2.8.18.1 Namespace ID (IS)

Refer to *user-defined table 0300 - Namespace ID* for suggested values.

2.8.18.2 Universal ID (ST)

The HD's second component, Universal ID (UID), is a string formatted according to the scheme defined by the third component, Universal ID type (UID type). The UID is intended to be unique over time within the UID type. It is rigorously defined. Each UID must belong to one of the specifically enumerated schemes for constructing UID's (defined by the UID type). The UID (second component) must follow the syntactic rules of the particular universal identifier scheme (defined by the third component).

2.8.18.3 Universal ID type (ID)

The third component governs the interpretation of the second component of the HD. If the third component is a known UID refer to *HL7 table 0301 - Universal ID type* for valid values, then the second component is a universal ID of that type.

Table 0301 - Universal ID type

Value	Description
DNS	An Internet dotted name. Either in ASCII or as integers
GUID	Same as UUID.
HCD	The CEN Healthcare Coding Scheme Designator. (Identifiers used in DICOM follow this assignment scheme.)
HL7	Reserved for future HL7 registration schemes
ISO	An International Standards Organization Object Identifier
L,M,N	These are reserved for locally defined coding schemes.
Random	Usually a base64 encoded string of random bits. The uniqueness depends on the length of the bits. Mail systems often generate ASCII string "unique names," from a combination of random bits and system names. Obviously, such identifiers will not be constrained to the base64 character set.
UUID	The DCE Universal Unique Identifier
x400	An X.400 MHS format identifier
x500	An X.500 directory name

Note: X400, X500, and DNS are not technically universally valid for all time. Names can be de-registered from an existing user and registered to a new user.

Examples:

```
1.2.34.4.1.5.1.5.1,1.13143143.131.3131.1^ISO

14344.14144321.4122344.14434.654^GUID

falcon.iupui.edu^DNS

40C983F09183B0295822009258A3290582^RANDOM

LAB1                Local use only: an HD that looks like an IS data
type.


PathLab^UCF.UC^L        A locally defined HD in which the middle
component is itself
                    structured.  This can be considered the combination
of
                    'PathLab' with the locally defined UID system "L".

LAB1^1.2.3.3.4.6.7^ISO An HD with an ISO "Object Identifier" as a
suffix, and a
                    locally defined system name.


^1.2.344.24.1.1.3^ISO  An HD consisting only of an ISO UID.
```

2.8.19 ID - coded value for HL7 defined tables

The value of such a field follows the formatting rules for an ST field except that it is drawn from a table of legal values. There shall be an HL7 table number associated with ID data types. Examples of ID fields include religion and sex. This data type should be used only for HL7 tables (see Section 2.6.7, "ID number"). The reverse is not true, since in some circumstances it is more appropriate to use the CE data type for HL7 tables.

2.8.20 IS - coded value for user-defined tables

The value of such a field follows the formatting rules for an ST field except that it is drawn from a site-defined (or user-defined) table of legal values. There shall be an HL7 table number associated with IS data types. An example of an IS field is the *Event reason code* defined in Section 3.3.1.4, "Event reason code." This data type should be used only for user-defined tables (see Section 2.6.7, "ID number"). The reverse is not true, since in some circumstances, it is more appropriate to use the CE data type for user-defined tables.

2.8.21 JCC - job code/class

```
Components:  <job code (IS)> ^ <job class (IS)>
```

2.8.21.1 Job code (IS)

This component contains the person's job code. Refer to *user-defined table 0327 - job code*.

2.8.21.2 Job class (IS)

This component contains the person's employee classification. *Refer to user-defined table 0329 - job class.*

2.8.22 MA - multiplexed array

```
Components:  <sample 1 from channel 1 (NM)> ^ <sample 1 from channel 2 (NM)> ^ <sample 1 from channel 3 (NM)>
              ...~<sample 2 from channel 1 (NM)> ^ <sample 2 from channel 2 (NM)> ^ <sample 2 from channel 3
             (NM)> ...~
              ...
```

This data type is used to represent channel-multiplexed waveform data, (e.g., the digitized values from an analog-to-digital converter or other digital data source). Refer to Chapter 7, Section 7.15.2, "MA - multiplexed array," for a complete description of this data type.

2.8.23 MO - money

```
Components:  <quantity (NM)> ^ <denomination (ID)>
```

2.8.23.1 Quantity (NM)

The first component is a quantity.

2.8.23.2 Denomination (ID)

The second component is the denomination in which the quantity is expressed. The values for the denomination component are those specified in ISO-4217. If the denomination is not specified, *MSH-17-country code* is used to determine the default.

Example:

```
|99.50^USD|
```

where USD is the ISO 4217 code for the U.S. American dollar.

2.8.24 NA - numeric array

This data type is used to represent a series (array) of numeric values, each one having a data type of NM. Refer to Chapter 7, Section 7.15.1, "NA - numeric array," for a complete description of this data type.

2.8.25 NM - numeric

A number represented as a series of ASCII numeric characters consisting of an optional leading sign (+ or -), the digits and an optional decimal point. In the absence of a sign, the number is assumed to be positive. If there is no decimal point the number is assumed to be an integer. Examples:

```
|999|
|-123.792|
```

Leading zeros, or trailing zeros after a decimal point, are not significant. For example, the following two values with different representations, "01.20" and "1.2", are identical. Except for the optional leading sign (+ or -) and the optional decimal point (.), no non-numeric ASCII characters are allowed. Thus, the value <12 should be encoded as a structured numeric (SN) (preferred) or as a string (ST) (allowed, but not preferred) data type.

2.8.26 PL - person location

```
Components:  <point of care  (IS )> ^ <room (IS )> ^ <bed (IS )> ^ <facility (HD)> ^ < location status  (IS )>
              ^ <person location type (IS)> ^ <building (IS )> ^ <floor (IS )> ^ <location description (ST)>
```

This data type is used to specify a patient location within a healthcare institution. Which components are valued depends on the needs of the site. It is most commonly used for specifying patient locations, but may refer to other types of persons within a healthcare setting.

2.8.26.1 Point of care (IS)

Conditional on person location type (e.g., nursing unit or department or clinic). After floor, most general patient location designation. Refer to *user-defined table 0302 - Point of care* for suggested values.

2.8.26.2 Room (IS)

Patient room. After nursing unit, most general person location designation. Refer to *user-defined table 0303 - Room* for suggested values.

2.8.26.3 Bed (IS)

Patient bed. After room, most general person location designation. Refer to *user-defined table 0304 - Bed* for suggested values.

2.8.26.4 Facility (HD)

Most general person location designation. (See Section 2.8.18, "HD - hierarchic designator").

2.8.26.5 Location status (IS)

Location (e.g., Bed) status. Refer to *user-defined table 0306 - Location status* for suggested values.

2.8.26.6 Person location type (IS)

Usually includes values such as nursing unit, department, clinic, SNF, physician's office. Refer to *user-defined table 0305 - Person location type* for suggested values.

2.8.26.7 Building (IS)

After facility, most general person location designation. Refer to *user-defined table 0307 - Building* for suggested values.

2.8.26.8 Floor (IS)

After building, most general person location designation. Refer to *user-defined table 0308 - Floor* for suggested values.

2.8.26.9 Location description (ST)

A free text description of the location.

Note	The actual order of components allows compatibility with previous versions of HL7. Without backward compatibility constraints, the hierarchical, structural order of components would be: <person location type (IS) > ^ <facility (HD)> ^ < building (IS)> ^ <floor (IS) > ^ <point of care (IS) > ^ <room (IS)> ^ <bed (IS)> ^

<location description (ST)> ^ < location status (IS)>.

2.8.27 PPN - performing person time stamp

```
Components: <ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^
            <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <source
            table (IS)> ^ <assigning authority (HD)> ^ <name type code(ID)>  ^ <identifier check digit (ST)>
            ^ <code identifying the check digit scheme employed (ID )> ^ <identifier type code (IS)> ^
            <assigning facility (HD)> ^ < date/time action performed (TS)>
```

This data type is the equivalent of an XCN data type joined with a TS data type. However, since HL7 does not support subcomponents in Version 2.3, the XCD data type has been flattened.

2.8.27.1 ID number (ST)

Coded ID according to a user-defined table, defined by the 8th component. If the first component is present, either the source table or the assigning authority must be valued.

2.8.27.2 Family name (ST)

2.8.27.3 Given name (ST)

2.8.27.4 Middle initial or name (ST)

2.8.27.5 Suffix (ST)

Used to specify a name suffix (e.g., Jr. Or III).

2.8.27.6 Prefix (ST)

Used to specify a name prefix (e.g., Dr.).

2.8.27.7 Degree (ST)

Used to specify an educational degree (e.g., MD).

2.8.27.8 Source table (ID)

Refer to *user-defined table 0207 - CN ID* for suggested values. Used to delineate the first component.

2.8.27.9 Assigning authority (HD)

In this version, an optional 9th component, the assigning authority (HD), has been added. It is an HD data type (see Section 2.8.18, "HD - hierarchic designator").

2.8.27.10 Name type code (ID)

A code that represents the type of name. Refer to *HL7 table 0200 - Name type* for valid values (see Section 2.8.48, "XPN - extended person name").

2.8.27.11 Identifier check digit (ST)

The check digit in this data type is not an add-on produced by the message processor. It is the check digit that is part of the identifying number used in the sending application. If the sending application does not include a self-generated check digit in the identifying number, this component should be valued null.

2.8.27.12 Code identifying the check digit scheme employed (ID)

Refer *to HL7 table 0060 - Check digit scheme* for valid values.

2.8.27.13 Identifier type code (ID)

A code corresponding to the type of identifier. In some cases, this code may be used as a qualifier to the "Assigning authority" component. Refer *to user-defined table 0203 - Identifier type* for suggested values.

2.8.27.14 Assigning facility (HD)

The place or location identifier where the identifier was first assigned to the patient. This component is not an inherent part of the identifier but rather part of the history of the identifier: as part of this data type, its existence is a convenience for certain intercommunicating systems.

2.8.27.15 Date/time action performed (TS)

This component describes when the activity was performed.

Note: If this field is not null, both the performing person and the time stamp must be valued

2.8.28 PN - person name

```
Components:  <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^ <suffix (e.g., JR or
              III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)>
```

Example:

```
|SMITH^JOHN^J^III^DR^PHD|
```

This data type includes multiple free text components. Each component is specified to be an HL7 ST data type. ***The maximum length of a PN field is 48 characters including component separators***. The sending system may send upper- and lowercase or all uppercase. The receiving system may convert to all uppercase if required.

2.8.28.1 Family name (ST)

2.8.28.2 Given name (ST)

2.8.28.3 Middle initial or name (ST)

2.8.28.4 Suffix (ST)

Used to specify a name suffix (e.g., Jr. or III).

2.8.28.5 Prefix (ST)

Used specify a name prefix (e.g., Dr.).

2.8.28.6 Degree (ST)

Used to specify an educational degree (e.g., MD).

2.8.28.6.1 **Internationalization note:** In countries using ideographic or syllabic (phonetic) character sets, it is sometimes necessary to send the name in one or both of these formats, as

well as an alphabetic format. The switching between the different character sets can be accomplished using a character set such as JIS X 0202 - ISO 2022 which provides an escape sequence for switching among different character sets and among single-byte and multi-byte character representations. When the name field is repeated, the different repetitions of the name may be represented by these different character sets. The details are as follows. (See also Section 2.9.2, "Escape sequences supporting multiple character sets for PN and XPN data types."

HL7 supports the following standards for Japanese characters:

JIS X 0201	for ISO-IR 13 (Japanese Katakana)
	for ISO-IR 14 (Japanese Romaji)
JIS X 0208	for ISO-IR 87 (Japanese Kanji, Hirigana and Katakana)
JIS X 0212	for ISO-IR 159 (supplementary Japanese Kanji)

HL7 supports the following standards for European characters:

ISO 8859 (1-9)	for ISO-IR 100, 101, 109, 110, 144,127, 126, 138 and 148.

Character sets are referenced in HL7 as ASCII, 8859/1..8859/2, JAS2020, and JIS X 0202. DICOM uses codes laid out in ISO 2375, of the form 'ISO-IR xxx'. HL7 supports this naming as well, to facilitate interoperability.

HL7 uses the Basic G0 Set of the International Reference Version of ISO 646:1990 (ISO IR-6) as the default character repertoire for character strings. This is a single-byte character set, identical to ASCII.

Each repetition of a PN or XPN field is assumed to begin with the default character set. If another character set is to be used, the HL7 defined escape sequence used to announce that character set must be at the beginning of the repetition, and the HL7 defined escape sequence used to start the default character set must be at the end of the repetition. Note also that several character sets may be intermixed within a single repetition as long as the repetition ends with a return to the default character set.

An application must specify which character sets it supports in its conformance statement and in the field *MSH-18-character-set*. It is assumed that the sending and receiving applications are aware of how to map character set names (i.e., ISO-IR xxx) to escape sequences.

For example, in many Japanese messages there is a mix of Romaji (i.e., Roman characters), Katakana (phonetic representation of foreign words), Hirigana (phonetic representation of Japanese words) and Kanji (pictographs). Such a message would require 4 character sets be specified in the MSH.

2.8.28.7 References for internationalization of name

1. "Understanding Japanese Information Processing" by Ken Lunde, O'Reilly Press

2. "DICOM Supplement 9 : Multi-Byte Character Set Support", ACR-NEMA

3. ANSI X3.4:1986 ASCII character set

4.	ISO 646:1990	Information Processing - ISO 7-bit coded character set for information interchange
5.	ISO/IEC 2022:1994	Information Technology - Character code structure and extension techniques
6.	ISO 2375:1986	Data Processing - Procedure for the registration of escape sequences
7.	ISO 6429:1990	Information Processing - Control functions for 7-bit and 8-bit coded character sets
8.	ISO 8859 (1-9)	Information Processing - 8-bit single-byte coded graphic character sets - parts 1-9
9.	ENV 41 503:1990	Information systems interconnection - European graphic character repertoires and their coding
10.	ENV 41 508:1990	Information systems interconnection - East European graphic character repertoires and their coding
11.	JIS X 0201-1976	Code for Information Exchange
12.	JIS X 0212-1990	Code of the supplementary Japanese Graphic Character set for information interchange
13.	JIS X 0208-1990	Code for the Japanese Graphic Character set for information interchange
14.	RFC 1468	Japanese Character Encoding for Internet Messages

This approach is in harmony with DICOM.

Character Repertoires supported by DICOM are defined in Part 5, section 62E1, of Supplement 9. It says, "Values that are text or character strings can be composed of Graphic and Control Characters. The Graphic Character set, independent of its encoding, is referred to as a Character Repertoire. Depending on the native context in which Application Entities wish to exchange data using the DICOM standard, different character repertoires will be used. The Character Repertoires supported by DICOM are defined in ISO 8859."

In addition, DICOM supports the following Character Repertoires for the Japanese Language :

JIS X 0201-1976 - Code for Information Exchange

JIS X 0208-1990 - Code for the Japanese Graphic Character set for information interchange

JIS X 0212-1990 - Code of the supplementary Japanese Graphic Character set for information interchange

2.8.29 PT - processing type

```
Components:  <processing ID (ID)> ^ <processing mode (ID)>
```

This data type indicates whether to process a message as defined in HL7 Application (level 7) Processing rules.

2.8.29.1 Processing ID (ID)

A value that defines whether the message is part of a production, training, or debugging system. Refer to *HL7 table 0103 - Processing ID* for valid values.

2.8.29.2 Processing mode (ID)

A value that defines whether the message is part of an archival process or an initial load. Refer to *HL7 table 0207 - Processing mode* for valid values.

2.8.30 QIP - query input parameter list

```
Components: <field name (ST) > ^ <value1 (ST) & value2 (ST) & value3 (ST) ...>
```

Definition: This field contains the list of parameter names and values to be passed to the stored procedure.

2.8.30.1 Field name (ST)

This component contains the field name.

Field naming conventions:

Fields are designated by the "@" symbol concatenated with the HL7 item number for the field. If the field is divided into components, the designation may be suffixed with ".nn," to identify a particular component (a suffix of ".3" indicates the third component of the field); otherwise, the whole field is assumed. If the field is further divided into subcomponents, the designation is suffixed with ".nn.mm," which identifies the component and subcomponent requested by relative position.

Site-specific fields may be used, provided that they begin with the letter **"Z."** In this case, site-specific item numbers must be defined that do not conflict with existing HL7 item numbers.

Values for this field are defined in the function-specific chapters of this specification.

Note:	If the "@" is being used as one of the delimiter characters defined in *MSH-2-encoding characters*, it must be "escaped ." (See Section 2.9.1," Formatting codes".)

2.8.30.2 Value1 & value2 & value3 (ST)

This component contains the field value or values in the form "value1& value2 & value3 …"

A single valued parameter contains only a single subcomponent in the second component: thus no subcomponent delimiters are needed (e.g., `<field name> ^ <value>`). A simple list of values (i.e., a one-dimensional array) may be passed instead of a single value by separating each value with the subcomponent delimiter: ,"`<field name>` ^ `<value1 & value2 &…>`"

2.8.31 QSC - query selection criteria

```
Components: <name of field (ST)> ^ <relational operator (ID)> ^ <value (ST)> ^ <relational conjunction (ID)>
```

Definition: This field indicates the conditions that qualify the rows to be returned in the query response. (This field conveys the same information as the "WHERE" clause in the corresponding SQL expression of the query, but is formatted differently.)

2.8.31.1 Name of field (ST)

The name of the field that is participating as a qualifier (usually the "key"). Refer to Section 2.8.30, "QIP - query input parameter list," for field naming conventions.

2.8.31.2 Relational operator (ID)

Refer to *HL7 table 0209 - Relational operator* for valid values.

Table 0209 - Relational operator

Relational operator	Value
EQ	Equal
NE	Not Equal
LT	Less than
GT	Greater than
LE	Less than or equal
GE	Greater than or equal
CT	Contains
GN	Generic

2.8.31.3 Value (ST)

The value to which the field will be compared.

2.8.31.4 Relational conjunction (ID)

Refer to *HL7 table 0102 - Relation conjunction* for valid values. The relational conjunction, defined as follows: If more than one comparison is to be made to select qualifying rows, a conjunction relates this repetition of the field to the next.

Table 0210 - Relational conjunction

Relational conjunction	Note
AND	Default
OR	

- When applied to strings, the relational operators LT, GT, LE, and GE imply an alphabetic comparison.
- A "generic" comparison selects a record for inclusion in the response when the beginning of the designated field matches the select string.
- Where a repeating field is specified as an operand, a match on any instance of that field qualifies the row for inclusion in the response message.

- AND takes precedence over OR. More sophisticated precedence rules require that the query be expressed as an embedded query language message or a stored procedure query message (see Section 2.19, "ENHANCED MODE QUERY MESSAGES," and also Sections 2.24.16, "EQL - embedded query language segment," and 2.24.20, "SPR - stored procedure request definition segment."

2.8.32 RCD - row column definition

```
Components: <HL7 item number (ST)> ^ <HL7 data type (ST)> ^ <maximum column width (NM)>
```

Definition: Each repetition of this field consists of three components:

2.8.32.1 HL7 item number (ST)

The HL7 item number, which identifies the field occupying the column. (Refer to Section 2.8.30, "QIP - query input parameter list," for item numbering conventions.)

2.8.32.2 HL7 data type (ST)

The two or three character HL7 data type, as defined in Section 2.8, "Data types."

2.8.32.3 Maximum column width (NM)

The maximum width of the column, as dictated by the responding system. (This may vary from the HL7-defined maximum field length.)

2.8.33 RI - repeat interval

```
Components:  <repeat pattern (IS)> ^ <explicit time interval (ST)>
```

Definition: This field contains the interval between repeating appointments. The default setting indicates that the appointment should occur once, when the component is not valued. The definition of this field is equivalent to the definition of the Interval component of the Quantity/Timing field given in Chapter 4, Section 4.4.2 "Interval component (CM)."

2.8.33.1 Repeat pattern (IS)

The first component is defined by *user-defined table 0335 - Repeat pattern*. See Section 4.4.2.1 "Repeat pattern," for further details.

2.8.33.2 Explicit time interval (ST)

The second component explicitly lists the actual times referenced by the code in the first subcomponent, in the following format: HHMM,HHMM,HHMM,…. This second subcomponent will be used to clarify the first subcomponent in cases where the actual administration times vary within an institution. See Section 4.4.2.2, "Explicit time interval subcomponent," for further details.

2.8.34 RP - reference pointer

```
Components:  <pointer (ST) > ^ < application ID (HD)> ^ <type of data (ID)> ^ <subtype (ID)>
```

This data type transmits information about data stored on another system. It contains a reference pointer that uniquely identifies the data on the other system, the identity of the other system, and the type of data.

2.8.34.1 Pointer (ST)

A unique key assigned by the system that stores the data. The key, which is an ST data type, is used to identify and access the data.

2.8.34.2 Application ID (HD)

```
Subcomponents: <namespace ID (IS)> & < universal ID (ST)> & <universal ID type (ID)>
```

A unique designator of the system that stores the data. It is an HD data type (See Section 2.8.18, "HD - hierarchic designator"). It is equivalent to the application ID of the placer or filler order number (see Chapter 4). Application ID's must be unique across a given HL7 implementation.

2.8.34.3 Type of data (ID)

An ID data type that declares the general type of data. Refer to *HL7 table 0191- Type of referenced data* for valid values.

Table 0191 - Type of referenced data

Value	Description
SI	Scanned image (HL7 v 2.2 only)
NS	Non-scanned image (HL7 v 2.2 only)
SD	Scanned document (HL7 v 2.2 only)
TX	Machine readable text document (HL7 v 2.2 only)
FT	Formatted text (HL7 v 2.2 only)
Image	Image data (HL7 v 2.3)
Audio	Audio data (HL7 v 2.3)
Application	Other application data, typically uninterpreted binary data (HL7 v 2.3)

2.8.34.4 Subtype (ID)

An ID data type declaring the format for the data of subcomponent <main type>. Refer to *HL7 table 0291 - Subtype of referenced data* for valid values.

Table 0291 - Subtype of referenced data

Value	Description
TIFF	TIFF image data
PICT	PICT format image data
DICOM	Digital Imaging and Communications in Medicine
FAX	Facsimile data
JOT	Electronic ink data (Jot 1.0 standard)
BASIC	ISDN PCM audio data
Octet-stream	Uninterpreted binary data
PostScript	PostScript program
JPEG	Joint Photographic Experts Group
GIF	Graphics Interchange Format
HTML	Hypertext Markup Language
RTF	Rich Text Format

2.8.34.5 Type-subtype combinations

Possible subtypes are specific to main types (though in principle the same subtype could be used for more than one main type), and so are defined under their main types.

Additional subtypes may be added to this Standard. In addition, private, non-standard subtypes may be defined by agreement between cooperating parties. All private, non-standard subtypes should begin with the letter **Z** to distinguish them from the standard subtypes.

2.8.34.5.1 Image subtypes

```
TIFF = TIFF image data
```

TIFF (Tagged Image File Format) is one of the common formats for scanned images. Its first version was developed in 1986 by Aldus Corporation as a standard for encoding scanned images. The official version of the TIFF standard is now maintained by Adobe Corporation. TIFF format is specified in the document "TIFF, Revision 6.0." Adobe Systems Incorporated, 1585 Charleston Road, P.O. Box 7900, Mountain View, CA 94039-7900.

415-961-4400

The subtype "TIFF" implies recognition of that trademark and all the rights it entails.

```
PICT = PICT format image data
```

PICT is one of the common formats for scanned images. PICT is a graphics format developed by Apple Computer, Inc., Cupertino, California. PICT format is officially defined in the book set "Inside Macintosh," published by Addison-Wesley Publishing Company, Reading, Massachusetts.

```
DICOM = the Digital Imaging and Communications in Medicine (DICOM) standard
```

DICOM is the format developed jointly by the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) as the standard for interchange of radiological images and ancillary data. It is standardized as NEMA PS3, and is available from:

NEMA

2101 L Street NW

Washington, DC 20037

DICOM specifies a complete communications standard, including a generic messaging service for two-way exchange of imaging-related information between applications, as well as transfer of the actual images. In HL7, the use of DICOM data is limited to images only.

Images in this subtype shall be encoded according to the Generic DICOM File Format defined in DICOM Part 10, Media Storage and File Format (NEMA PS3.10). This shall be in accordance with the Image Information Object Definitions of DICOM Part 3 (NEMA PS3.3), Data Structure and Semantics of DICOM Part 5 (NEMA PS3.5), and the Data Dictionary of DICOM Part 6 (NEMA PS3.6).

The Generic DICOM File Format consists of two parts: a DICOM File Meta Information Header, immediately followed by a DICOM Data Set. The DICOM Data Set contains the image or images specified according to DICOM Part 10. The DICOM File Meta Information Header contains, among other information, a Transfer Syntax UID (Unique Identifier) which completely specifies the encoding of the Data Set according to DICOM Part 5. This encoding defines big endian vs. little endian byte ordering, as well as image compression via the JPEG (Joint Photographics Experts Group) standard (ISO/IS 10918-1 and 10918-2). The transfer syntax of the File Meta Information Header itself is little endian byte ordered, as required by DICOM Part 10.

```
FAX = facsimile data
```

Facsimile data as specified by CCITT standards F1.60, F1.80, F1.82, and F1.84.

```
Jot = electronic ink data, as specified by the Jot 1.0 standard
```

The JOT standard, proposed jointly by Slate Corporation, Microsoft, Apple, Lotus, GO, and General Magic, allows handwritten notes, sketches, signatures and other free-form written data to be transmitted. It is the standard by which portable pen computers or workstations equipped with stylus-input tablets can represent and exchange information.

It represents electronic ink as a series of stylus strokes, and therefore contains necessary information for potential automatic handwriting recognition, which would be lost if converted to other image representations. It may, however, be readily converted to another image representation for purposes of printing or display.

The JOT 1.0 standard is available from:

Software Publishers Association

1730 M Street Northwest, Suite 700

Washington, DC 20036-4510

202-452-1600

2.8.34.5.2 Audio subtypes

```
basic = ISDN PCM audio data
```

Telephone quality audio data, encoded as 8-bit ISDN mu-law Pulse Code Modulation sampled at 8 kHz, according to CCITT Fascicle III.4, Recommendation G.711. This subtype may be used for voice mail messages as well as voice dictation.

2.8.34.5.3 Application subtypes

```
octet-stream = uninterpreted binary data
```

This subtype is for binary data which has none of the other standard formats as given by Section 2.8.34.3, "Type of data (ID)." Its interpretation by the system utilizing the data must be mutually agreed upon by sending and receiving parties.

```
PostScript = PostScript program
```

A PostScript language program typically representing a formatted document for printing on a PostScript printer, or for display on a computer screen via a PostScript interpreter.

PostScript consists of the original specification, PostScript level 1, described in "PostScript Language Reference Manual," Addison-Wesley, 1985, and a more advanced variant, PostScript level 2, described in "PostScript Language Reference Manual," Addison-Wesley, Second Edition, 1990. PostScript is a registered trademark of Adobe Systems, Inc. Use of the subtype "PostScript" implies recognition of that trademark and all the rights it entails.

Other types may be added as needed.

Example:

```
|1234A321634BC^EFC^SD|
```

2.8.35 SCV - scheduling class value pair

```
Components:  <parameter class (IS)> ^ <parameter value (ST)>
```

For use only with the scheduling chapter.

Definition: This field is used to communicate parameters and preferences to the filler application regarding the selection of an appropriate *time slot, resource, location, or filler override criterion* for an appointment.

2.8.35.1 Parameter class (IS)

The first component of this field is a code identifying the parameter or preference being passed to the filler application.

2.8.35.2 Parameter value (IS)

The second component is the actual data value for that parameter.

For example, if a filler application allows preference parameters to be passed to specify a preferred start time, a preferred end time, and preferred days of the week for the appointment, it may define the following parameter class codes and valid data sets.

User-defined Table 0294 - Time selection criteria parameter class codes

Parameter Class	Description: Valid Values
PREFSTART	The preferred start time for the appointment request, service or resource. Any legal time specification in the format HHMM, using 24-hour clock notation
PREFEND	The preferred end time for the appointment request, service or resource. Any legal time specification in the format HHMM, using 24-hour clock notation
MON	An indicator that Monday is or is not preferred for the day on which the appointment will occur. OK = Preferred appointment day NO = Day is not preferred
TUE	An indicator that Tuesday is or is not preferred for the day on which the appointment will occur. OK = Preferred appointment day NO = Day is not preferred
WED	An indicator that Wednesday is or is not preferred for the day on which the appointment will occur. OK = Preferred appointment day NO = Day is not preferred
THU	An indicator that Thursday is or is not preferred for the day on which the appointment will occur. OK = Preferred appointment day NO = Day is not preferred
FRI	An indicator that Friday is or is not preferred for the day on which the appointment will occur. OK = Preferred appointment day NO = Day is not preferred
SAT	An indicator that Saturday is or is not preferred for the day on which the appointment will occur. OK = Preferred appointment day

	NO = Day is not preferred
SUN	An indicator that Sunday is or is not preferred for the day on which the appointment will occur. OK = Preferred appointment day NO = Day is not preferred

2.8.36 SI - sequence ID

A non-negative integer in the form of an NM field. The uses of this data type are defined in the chapters defining the segments and messages in which it appears.

2.8.37 SN - structured numeric

```
Components:  <comparator (ST)> ^ <num1 (NM)> ^ <separator/suffix (ST)> ^ <num2 (NM)>
```

The structured numeric data type is used to unambiguously express numeric clinical results along with qualifications. This enables receiving systems to store the components separately, and facilitates the use of numeric database queries. The corresponding sets of values indicated with the <comparator> and <separator/suffix> components are intended to be the authoritative and complete set of values. If additional values are needed for the <comparator> and <separator/suffix> components, they should be submitted to HL7 for inclusion in the Standard.

If <num1> and <num2> are both non-null, then the separator/suffix must be non-null. If the separator is "-", the data range is inclusive; e.g., <num1> - <num2> defines a range of numbers *x*, such that: <num1> <=*x*<= <num2>.

2.8.37.1 <comparator> = ">" or "<" or ">=" or "<=" or "=" or "<>"

Defined as greater than, less than, greater than or equal, less than or equal, equal, and not equal, respectively. If this component is not valued, it defaults to equal ("=").

2.8.37.2 <num1> (NM)

A number.

2.8.37.3 <num2> (NM)

A number or null depending on the measurement.

2.8.37.4 <separator/suffix> (ST)

"-" or "+" or "/" or "." or ":"

Examples:

```
                |>^100|             (greater than 100)
                |^100^-^200|     (equal to range of 100 through 200)
                |^1^:^228|          (ratio of 1 to 128, e.g., the results of a
serological test)
                |^2^+|              (categorical response, e.g., occult blood
positivity)
```

2.8.38 ST - string data

String data is left justified with trailing blanks optional. Any displayable (printable) ACSII characters (hexadecimal values between 20 and 7E, inclusive, or ASCII decimal values between 32 and 126), except the defined delimiter characters. Example:

```
|almost any data at all|
```

To include any HL7 delimiter character (except the segment terminator) within a string data field, use the appropriate HL7 escape sequence (see Section 2.9.1, "Formatting codes").

Usage note: the ST data type is intended for short strings (e.g., less than 200 characters). For longer strings the TX or FT data types should be used (see Sections 2.8.43, "TX - text data," or 2.8.17, "FT - formatted text data").

2.8.39 TM - time

```
Format: HH[MM[SS[.S[S[S[S]]]]]][+/-ZZZZ]
```

In prior versions of HL7, this data type was always specified to be in the format HHMM[SS[.SSSS]][+/-ZZZZ] using a 24 hour clock notation. In the current and future versions, the precision of a time may be expressed by limiting the number of digits used with the format specification as shown above. By site-specific agreement, HHMM[SS[.SSSS]][+/-ZZZZ] may be used where backward compatibility must be maintained.

Thus, HH is used to specify a precision of "hour," HHMM is used to specify a precision of "minute," HHMMSS is used to specify a precision of seconds, and HHMMSS.SSSS is used to specify a precision of ten-thousandths of a second.

In each of these cases, the time zone is an optional component. The fractional seconds could be sent by a transmitter who requires greater precision than whole seconds. Fractional representations of minutes, hours or other higher orders units of time are not permitted.

The time zone of the sender may be sent optionally as an offset from the coordinated universal time (previously known as Greenwich Mean Time). Where the time zone is not present in a particular TM field but is included as part of the date/time field in the MSH segment, the MSH value will be used as the default time zone. Otherwise, the time is understood to refer to the local time of the sender. Midnight is represented as 0000. Examples:

```
|235959+1100|       1 second before midnight in a time zone eleven
                  hours ahead of Universal Coordinated Time (i.e.,
                   east of Greenwich).

|0800|              Eight AM, local time of the sender.

|093544.2312|       44.2312 seconds after Nine thirty-five AM, local
time
                   of sender.

|13|              1pm (with a precision of hours), local time of sender.
```

2.8.40 TN - telephone number

For use in the United States and conforming countries, the telephone number is always in the form:

```
Format:  [NN] [(999)]999-9999[X99999][B99999][C any text]
```

The optional first two digits are the country code. The optional **X** portion gives an extension. The optional **B** portion gives a beeper code. The optional **C** portion may be used for comments like, **After 6:00**. While no explicit limit is placed on the text field, receiving systems may be expected to truncate values that are more than 10 characters long. To accommodate the variability of institutional phone systems, the length of the extension and beeper numbers may be extended by local agreement. Examples:

```
|(415)925-0121X305|

|234-4532CWEEKENDS|
```

2.8.41 TQ - timing quantity

Describes when a service should be performed and how frequently. See Chapter 4 (Section 4.4, "QUANTITY/TIMING (TQ) DEFINITION") for a complete description of this data type.

2.8.42 TS - time stamp

```
Format:  YYYY[MM[DD[HHMM[SS[.S[S[S[S]]]]]]]][+/-ZZZZ]^<degree of precision>
```

Contains the exact time of an event, including the date and time. The date portion of a time stamp follows the rules of a date field and the time portion follows the rules of a time field. The specific data representations used in the HL7 encoding rules are compatible with ISO 8824-1987(E).

In prior versions of HL7, an optional second component indicates the degree of precision of the time stamp (Y = year, L = month, D = day, H = hour, M = minute, S = second). This optional second component is retained only for purposes of backward compatibility.

By site-specific agreement, YYYYMMDD[HHMM[SS[.S[S[S[S]]]]]][+/-ZZZZ]^<degree of precision> may be used where backward compatibility must be maintained.

In the current and future versions of HL7, the precision is indicated by limiting the number of digits used, unless the optional second component is present. Thus, YYYY is used to specify a precision of "year," YYYYMM specifies a precision of "month," YYYYMMDD specifies a precision of "day," YYYYMMDDHH is used to specify a precision of "hour," YYYYMMDDHHMM is used to specify a precision of "minute," YYYYMMDDHHMMSS is used to specify a precision of seconds, and YYYYMMDDHHMMSS.SSSS is used to specify a precision of ten thousandths of a second. In each of these cases, the time zone is an optional component. Maximum length of the time stamp is 26. Examples:

```
|19760704010159-0600|  1:01:59 on July 4, 1976 in the Eastern
                       Standard Time zone.

|19760704010159-0500|  1:01:59 on July 4, 1976 in the Eastern
                       Daylight Saving Time zone.

|198807050000|         Midnight of the night extending from July 4 to
                       July 5, 1988 in the local time zone of the sender.

|19880705|             Same as prior example, but precision extends
                       only to the day.  Could be used for a
                       birthdate, if the time of birth is unknown.
```

The HL7 Standard strongly recommends that all systems routinely send the time zone offset but does not require it. All HL7 systems are required to accept the time zone offset, but its implementation is application specific. For many applications the time of interest is the local time of the sender. For example, an application in the Eastern Standard Time zone receiving notification of an admission that takes place at 11:00 PM in San Francisco on December 11 would prefer to treat the admission as having occurred on December 11 rather than advancing the date to December 12.

One exception to this rule would be a clinical system that processed patient data collected in a clinic and a nearby hospital that happens to be in a different time zone. Such applications may choose to convert the data to a common representation. Similar concerns apply to the transitions to and from daylight saving time. HL7 supports such requirements by requiring that the time zone information be present when the information is sent. It does not, however, specify which of the treatments discussed here will be applied by the receiving system.

2.8.43 TX - text data

String data meant for user display (on a terminal or printer). Such data would not necessarily be left justified since leading spaces may contribute greatly to the clarity of the presentation to the user. Because this type of data is intended for display, it may contain certain escape character sequences designed to control the display. Escape sequence formatting is defined later in this chapter in Section 2.9, "Use of escape sequences in text fields." Leading spaces should be included. Trailing spaces should be removed. Example:

```
|  leading spaces are allowed.|
```

Since TX data is intended for display purposes, the repeat delimiter, when used with a TX data field, implies a series of repeating lines to be displayed on a printer or terminal. Therefore, the repeat delimiters are regarded as paragraph terminators or hard carriage returns (e.g., they would display as though a CR/LF were inserted in the text (DOS type system) or as though a LF were inserted into the text (UNIX style system)).

A receiving system would word-wrap the text between repeat delimiters in order to fit it into an arbitrarily sized display window but start any line beginning with a repeat delimiter on a new line.

Usage note: ***the maximum length of a TX data field is 64K.***

2.8.44 VH - visiting hours

```
Components:  <start day range (ID)> ^ <end day range (ID)> ^ <start hour range (TM)> ^ <end hour range (TM)>
```

Definition: This data type contains the hours when a patient location is open for visiting. Refer to *HL7 table 0267 - Days of the week* for valid values for the first two components.

2.8.44.1 Start day range (ID)

Starting day of visiting hours range. See *HL7 table 0267 - Days of the week* for values.

2.8.44.2 End day range (ID)

Ending day of visiting hours range. Starting day of visiting hours range. *See HL7 table 0267 - Days of the week* for values

Table 0267 - Days of the Week

Value	Description
SAT	Saturday
SUN	Sunday
MON	Monday
TUE	Tuesday
WED	Wednesday
THU	Thursday
FRI	Friday

2.8.44.3 Start hour range (TM)

Starting hour on starting day of visiting hours range (see first component, 2.8.44.1, “Start day range”).

2.8.44.4 End hour range (TM)

Ending hour on ending day of visiting hours range (see second component, 2.8.44.2, “End day range”).

2.8.45 XAD - extended address

```
Components:  <street address (ST)> ^ <other designation (ST)> ^ <city (ST)> ^ <state or province (ST)> ^ <zip
              or postal code(ST)> ^ <country (ID)> ^ < address type (ID)> ^ <other geographic designation
              (ST)>^ <county/parish code (IS)> ^ <census tract (IS)>
```

Example:

```
|1234 Easy St.^Ste. 123^San Francisco^CA^95123^USA^B^^SF^|
```

2.8.45.1 Street address (ST)

The street or mailing address of a person or institution.

2.8.45.2 Other designation (ST)

Second line of address. In general, it qualifies address. Examples: Suite 555 or Fourth Floor.

2.8.45.3 City (ST)

2.8.45.4 State or province (ST)

State or province should be represented by the official postal service codes for that country.

2.8.45.5 Zip or postal code (ST)

Zip or postal codes should be represented by the official codes for that country. In the US, the zip code takes the form 99999[-9999], while the Canadian postal code takes the form A9A-9A9.

2.8.45.6 Country (ID)

Defines the country of the address. ISO 3166 provides a list of country codes that may be used.

2.8.45.7 Address type (ID)

Address type is optional and defined by *HL7 table 0190 - Address type*.

2.8.45.8 Other geographic designation (ST)

Other geographic designation includes country, bioregion, SMSA, etc.

2.8.45.9 County/parish code (IS)

A code that represents the county in which the specified address resides. Refer to *user-defined table 0289 - County/parish*. When this component is used to represent the county (or parish), component 8 "other geographic designation" should not duplicate it (i.e., the use of "other geographic designation" to represent the county is allowed only for the purpose of backward compatibility, and should be discouraged in this and future versions of HL7).

Allowable values: codes defined by government.

2.8.45.10 Census tract (IS)

A code that represents the census track in which the specified address resides. Refer to *user-defined table 0288 - Census tract*.

Allowable Values: codes defined by government.

2.8.46 XCN - extended composite ID number and name for persons

```
Components: <ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^
              <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <source
              table (IS)> ^ <assigning authority (HD)> ^ <name type code(ID)>  ^ <identifier check digit (ST)>
              ^ <code identifying the check digit scheme employed (ID )> ^ <identifier type code (IS)> ^
              <assigning facility (HD)>

Subcomponents of assigning authority:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>

Subcomponents of assigning facility:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>
```

Example:

```
|1234567^Smith^John^J^III^DR^PHD^ADT01^^L^4^M11^MR|
```

2.8.46.1 ID number (ST)

Coded ID according to a user-defined table, defined by the 8th component. If the first component is present, either the source table or the assigning authority must be valued.

2.8.46.2 Family name (ST)

2.8.46.3 Given name (ST)

2.8.46.4 Middle initial or name (ST)

2.8.46.5 Suffix (ST)

Used to specify a name suffix (e.g., Jr. or III).

2.8.46.6 Prefix (ST)

Used to specify a name prefix (e.g., Dr.).

2.8.46.7 Degree (ST)

Used to specify an educational degree (e.g., MD).

2.8.46.8 Source table (IS)

Refer to *user-defined table 0207 - CN ID source* for suggested values. Used to delineate the first component.

2.8.46.9 Assigning authority (HD)

In this version an optional 9th component, the assigning authority (HD), has been added. It is an HD data type (see Section 2.8.18, "HD - hierarchic designator").

2.8.46.10 Name type code (ID)

A code that represents the type of name. Refer to *HL7 table 0200 - Name type* for valid values (see Section 2.8.48, "XPN - extended person name").

2.8.46.11 Identifier check digit (ST)

The check digit in this data type is not an add-on produced by the message processor. It is the check digit that is part of the identifying number used in the sending application. If the sending application does not include a self-generated check digit in the identifying number, this component should be valued null.

2.8.46.12 Code identifying the check digit scheme employed (ID)

Refer *to HL7 table 0060 - Check digit scheme* for valid values.

2.8.46.13 Identifier type code (IS)

A code corresponding to the type of identifier. In some cases, this code may be used as a qualifier to the "Assigning authority" component. Refer *to user-defined table 0203 - Identifier type* for suggested values.

2.8.46.14 Assigning facility (HD)

The place or location identifier where the identifier was first assigned to the patient. This component is not an inherent part of the identifier but rather part of the history of the identifier: as part of this data type, its existence is a convenience for certain intercommunicating systems.

2.8.47 XON - extended composite name and identification number for organizations

```
Components:  <organization name (ST)> ^ <organization name type code (IS)> ^ <ID Number (NM)> ^ <check digit
             (NM)> ^ <code identifying the check digit scheme employed (ID)> ^ <assigning authority  (HD)> ^
             <identifier type code (IS)> ^ <assigning facility ID (HD)>

Subcomponents of assigning authority:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>

Subcomponents of assigning facility:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>
```

Example:

```
|HL7 Health Center^L^6^M11^HCFA|
```

2.8.47.1 Organization name (ST)

The name of the specified organization.

2.8.47.2 Organization name type code (IS)

A code that represents the type of name i.e., legal name, display name. Refer to *user-defined table 0204 - Organizational name type* for suggested values.

User-defined table 0204 - Organizational name type

Value	Description
A	Alias Name
L	Legal Name
D	Display Name
SL	Stock Exchange Listing Name

2.8.47.3 ID number (NM)

2.8.47.4 Check digit (NM)

The check digit in this data type is not an add-on produced by the message processor. It is the check digit that is part of the identifying number used in the sending application. If the sending application does not include a self-generated check digit in the identifying number, this component should be valued null.

2.8.47.5 Code identifying the check digit scheme employed (ID)

The check digit scheme codes are defined in *HL7 table 0061 - Check digit scheme*.

2.8.47.6 Assigning authority (HD)

The assigning authority is a unique name of the system that creates the data. It is an HD data type. It is equivalent to the application ID of the placer or filler order number (see Chapter 4). Assigning authorities are unique across a given HL7 implementation.

2.8.47.7 Identifier type code (IS)

A code corresponding to the type of identifier. In some cases, this code may be used as a qualifier to the "Assigning authority" component. Refer to *user-defined table 0203 - Identifier type* for suggested values.

2.8.47.8 Assigning facility ID (HD)

The assigning facility is a unique identifier of the system that creates the data. It is an HD data type. It is equivalent to the application ID of the placer or filler order number (see Chapter 4). Assigning authorities are unique across a given HL7 implementation.

2.8.48 XPN - extended person name

```
Components:  <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^ <suffix (e.g., JR or
              III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <name type code (ID) > ^
              <name representation code (ID)>
```

Example:

```
|Smith^John^J^III^DR^PHD^L|
```

2.8.48.1 Family name (ST)

2.8.48.2 Given name (ST)

2.8.48.3 Middle initial or name (ST)

2.8.48.4 Suffix (ST)

Used to specify a name suffix (e.g., Jr. or III).

2.8.48.5 Prefix (ST)

Used to specify a name prefix (e.g., Dr.).

2.8.48.6 Degree (ST)

Used to specify an educational degree (e.g., MD).

2.8.48.7 Name type code (ID)

A code that represents the type of name. Refer to *HL7 table 0200 - Name type* for valid values.

Table 0200 - Name type

Value	Description
A	Alias Name
L	Legal Name
D	Display Name
M	Maiden Name
C	Adopted Name

Note:	The legal name is the same as the current married name.

2.8.48.8 Name representation code (ID)

Table 4000 - Name representation

Value	Description
I	Ideographic (i.e., Kanji)
A	Alphabetic (i.e., Default or some single-byte)
P	Phonetic (i.e., ASCII, Katakana, Hirigana, etc.)

In general this component provides an indication of the representation provided by the data item. It does not necessarily specify the character sets used. Thus, even though the representation might provide an indication of what to expect, the sender is still free to encode the contents using whatever character set is desired. This component provides only hints for the receiver, so it can make choices regarding what it has been sent and what it is capable of displaying.

2.8.49 XTN - extended telecommunication number

```
Components:  [NNN] [(999)]999-9999 [X99999] [B99999] [C any text] ^ <telecommunication use code (ID)> ^
              <telecommunication equipment type (ID)> ^ <email address (ST)> ^ <country code (NM)> ^
              <area/city code (NM)> ^ <phone number (NM)> ^ <extension (NM)> ^ <any text (ST)>
```

Example:

```
(415)555-3210^ORN^FX^
```

2.8.49.1 [(999)] 999-9999 [X99999] [C any text]

Defined as the TN data type (see Section 2.8.40, “TN - telephone number”), except that the length of the country access code has been increased to three.

2.8.49.2 Telecommunication use code (ID)

A code that represents a specific use of a telecommunication number. Refer *to HL7 table 0201 - Telecommunication use code* for valid values.

Table 0201 - Telecommunication use code

Value	Description
PRN	Primary Residence Number
ORN	Other Residence Number
WPN	Work Number
VHN	Vacation Home Number
ASN	Answering Service Number
EMR	Emergency Number
NET	Network (email) Address
BPN	Beeper Number

2.8.49.3 Telecommunication equipment type (ID)

A code that represents the type of telecommunication equipment. Refer *to HL7 table 0202 - Telecommunication equipment type* for valid values.

Table 0202 - Telecommunication equipment type

Value	Description
PH	Telephone
FX	Fax
MD	Modem
CP	Cellular Phone
BP	Beeper
Internet	Internet Address: Use Only If Telecommunication Use Code Is NET
X.400	X.400 email address: Use Only If Telecommunication Use Code Is NET

2.8.49.4 Email address (ST)

Internationalization note: To make this data type interoperate with CEN's Telecommunication data attribute group, we allow use of the second component for email addresses. The presence of an email address is specified by the addition of the value *NET* to the Phone Use Code table, and the type of Internet address is specified with the values *Internet* and *X.400* to the Phone Equipment Type table. When used for an Internet address, the first component of the XTN data type will be null. If the @-sign is being used as a subcomponent delimiter, the HL7 subcomponent escape sequence may be used when encoding an Internet address (see Section 2.9.1., "Formatting codes").

Note: Components five through nine reiterate the basic function of the first component in a delimited form that allows the expression of both local and international telephone numbers. In Version 2.3, the recommended form for the telephone number is to use the delimited form rather than the unstructured form supported by the first component (which is left in for backward compatibility only).

2.8.49.5 Country code (NM)

2.8.49.6 Area/city code (NM)

2.8.49.7 Phone number (NM)

2.8.49.8 Extension (NM)

2.8.49.9 Any text (ST)

2.9 USE OF ESCAPE SEQUENCES IN TEXT FIELDS

2.9.1 Formatting codes

When a field of type TX, FT, or CF is being encoded, the escape character may be used to signal certain special characteristics of portions of the text field. The escape character is whatever display ASCII character is specified in the Escape Character component of *MSH-2-encoding characters*. For purposes of this section, the character \ will be used to represent the character so designated in a message. An **escape sequence** consists of the escape character followed by an escape code ID of one character, zero (0) or more data characters, and another occurrence of the escape character. The following escape sequences are defined:

\H\	start highlighting
\N\	normal text (end highlighting)
\F\	field separator

\S\	component separator
\T\	subcomponent separator
\R\	repetition separator
\E\	escape character
\Xdddd...\	hexadecimal data
\Zdddd...\	locally defined escape sequence

The **escape sequences** for field separator, component separator, subcomponent separator, repetition separator, and escape character are also valid within an ST data field.

No escape sequence may contain a nested escape sequence.

2.9.2 Escape sequences supporting multiple character sets for PN and XPN data types

The following HL7 escape sequences are defined to support multiple character sets for fields of the PN and XPN data type. They allow HL7 parsers to use escape codes (defined in the standards used below), without breaking, and without being non-conformant, to the HL7 escape paradigm defined in this section.

\Cxxyy\ single-byte character set escape sequence with two hexadecimal values, xx and yy, that indicate the escape sequence defined for one of the character repertoires supported for the current message (i.e., ISO-IR xxx).

\Mxxyyzz\ multi-byte character set escape sequence with three hexadecimal values, xx, yy and zz. zz is optional.

Common character set escape sequences include the following which are defined in the standards mentioned:

Single-byte character sets:

\C2842\	ISO-IR 6 G0 (ISO 646 : ASCII)
\C2D41\	ISO-IR 100 (ISO 8859 : Latin Alphabet 1)
\C2D42\	ISO-IR 101 (ISO 8859 : Latin Alphabet 2)
\C2D43\	ISO-IR 109 (ISO 8859 : Latin Alphabet 3)
\C2D44\	ISO-IR 110 (ISO 8859 : Latin Alphabet 4)
\C2D4C\	ISO-IR 144 (ISO 8859 : Cyrillic)
\C2D47\	ISO-IR 127 (ISO 8859 : Arabic)
\C2D46\	ISO-IR 126 (ISO 8859 : Greek)
\C2D48\	ISO-IR 138 (ISO 8859 : Hebrew)
\C2D4D\	ISO-IR 148 (ISO 8859 : Latin Alphabet 5)
\C284A\	ISO-IR 14 (JIS X 0201 -1976: Romaji)
\C2949\	ISO-IR 13 (JIS X 0201 : Katakana)

Multi-byte codes :

\M2442\	ISO-IR 87 (JIS X 0208 : Kanji, hirigana and katakana)
\M242844\	ISO-IR 159 (JIS X 0212 : Supplementary Kanji)

2.9.3 Highlighting

In designating highlighting, the sending application is indicating that the characters that follow somehow should be made to stand out, but leaving the method of doing so to the receiving application. Depending on device characteristics and application style considerations, the receiving application may choose reverse video, boldface, underlining, blink, an alternate color or another means of highlighting the displayed data. For example the message fragment:

```
DSP|     TOTAL CHOLESTEROL    \H\240*\N\   [90 - 200]
```

might cause the following data to appear on a screen or report:

```
TOTAL CHOLESTEROL        240*    [90 - 200]
```

whereas another system may choose to show the 240* in red.

2.9.4 Special character

The special character escape sequences (\F\, \S\, \R\, \T\, and \E\) allow the corresponding characters to be included in the data in a text field, though the actual characters are reserved. For example, the message fragment

```
DSP|  TOTAL CHOLESTEROL     180  \F\90 - 200\F\
DSP|  \S\----------------\S\
```

would cause the following information to be displayed, given suitable assignment of separators:

```
TOTAL CHOLESTEROL     180  |90 - 200|
^----------------^
```

2.9.5 Hexadecimal

When the hexadecimal escape sequence (\Xdddd...\) is used the X should be followed by 1 or more pairs of hexadecimal digits (0, 1, . . . , 9, A, . . . , F). Consecutive pairs of the hexadecimal digits represent 8-bit binary values. The interpretation of the data is entirely left to an agreement between the sending and receiving applications that is beyond the scope of this Standard.

2.9.6 Formatted text

If the field is of the formatted text (FT) data type, formatting commands also may be surrounded by the escape character. Each command begins with the **.** (period) character. The following formatting commands are available:

.sp <number>	End current output line and skip <number> vertical spaces. <number> is a positive integer or absent. If <number> is absent, skip one space. The horizontal character position remains unchanged. Note that for purposes of compatibility with previous versions of HL7, "^\.sp\" is equivalent to "\.br\."
.br	Begin new output line. Set the horizontal position to the current left margin and

	increment the vertical position by 1.
.fi	Begin word wrap or fill mode. This is the default state. It can be changed to a no-wrap mode using the .nf command.
.nf	Begin no-wrap mode.
.in <number>	Indent <number> of spaces, where <number> is a positive or negative integer. This command cannot appear after the first printable character of a line.
.ti <number>	Temporarily indent <number> of spaces where number is a positive or negative integer. This command cannot appear after the first printable character of a line.
.sk < number>	Skip <number> spaces to the right.
.ce	End current output line and center the next line.

The component separator that marks each line defines the extent of the temporary indent command (.ti), and the beginning of each line in the no-wrap mode (.nf). Examples of formatting instructions that are NOT included in this data type include: width of display, position on page or screen, and type of output devices.

Figure 2-3 is an example of the FT data type from a radiology impression section of a radiology report:

Figure 2-3. Formatted text as transmitted

```
|\.in+4\\.ti-4\ 1.  The cardiomediastinal silhouette is now within normal
limits.^\.sp\\.ti-4\ 2.   Lung fields show minimal ground glass
appearance.^\.sp\\.ti-4\ 3.  A loop of colon visible in the left upper
quadrant is distinctly abnormal with the appearance of mucosal effacement
suggesting colitis.\.in-4\|
```

Figure 2-4 shows one way of presenting the data in *Figure 2-3*. The receiving system can create many other interpretations by varying the right margin.

Figure 2-4. Formatted text in one possible presentation

```
1.  The cardiomediastinal silhouette is now within normal limits.

2.  Lung fields show minimal ground glass appearance.

3.  A loop of colon visible in the left upper quadrant is distinctly abnormal with the appearance of
    mucosal effacement suggesting colitis.
```

2.9.7 Local

When the local escape sequence (\Zdddd...\) is used the Z should be followed by characters that are valid in a TX field. The interpretation of the data is entirely left to an agreement between the sending and receiving applications that is beyond the scope of this Standard.

2.10 MESSAGE CONSTRUCTION RULES

Note: These message construction rules define the standard HL7 encoding rules, creating variable length delimited messages. Although only one set of encoding rules is defined as a standard in HL7 Version 2.3, other encoding rules are possible (but since they are non-standard, they may only be used by a site-specific agreement).

Step 1 Construct the segments in the order defined for the message. Each message is constructed as follows:

a) the first three characters are the segment ID code

b) each data field in sequence is inserted in the segment in the following manner:

1) a field separator is placed in the segment

2) if the value is not present, no further characters are required

3) if the value is present, but null, the characters "" (two consecutive double quotation marks) are placed in the field

4) otherwise, place the characters of the value in the segment. As many characters can be included as the maximum defined for the data field. It is not necessary, and is undesirable, to pad fields to fixed lengths. Padding to fixed lengths is permitted. Encode the individual data fields as shown in Section 2.8, "DATA TYPES."

5) if the field definition calls for a field to be broken into components, the following rules are used:

i) if more than one component is included they are separated by the component separator

ii) components that are present but null are represented by the characters ""

iii) components that are not present are treated by including no characters in the component

iv) components that are not present at the end of a field need not be represented by component separators. For example, the two data fields are equivalent:

```
|ABC^DEF^^| and |ABC^DEF|.
```

6) if the component definition calls for a component to be broken into subcomponents, the following rules are used:

i) if more than one subcomponent is included they are separated by the subcomponent separator

II) subcomponents that are present but null are represented by the characters ""

iii) subcomponents that are not present are treated by including no characters in the subcomponent

iv) subcomponents that are not present at the end of a component need not be represented by subcomponent separators. For example, the two data components are equivalent:

```
^XXX&YYY&&^ and ^XXX&YYY^.
```

7) if the field definition permits repetition of a field, the following rules are used, the repetition separator is used only if more than one occurrence is transmitted and is placed between occurrences. (If three occurrences are transmitted, two repetition separators are used.) In the example below, two occurrences of telephone number are being sent:

```
|234-7120~599-1288B1234|
```

c) repeat Step 1b while there are any fields present to be sent. If all the data fields remaining in the segment definition are not present there is no requirement to include any more delimiters.

d) end each segment with an ASCII carriage return character

Step 2 Repeat Step 1 until all segments have been generated.

The following rules apply to receiving HL7 messages and converting their contents to data values:

a) ignore segments, fields, components, subcomponents, and extra repetitions of a field that are present but were not expected

b) treat segments that were expected but are not present as consisting entirely of fields that are not present

c) treat fields and components that are expected but were not included in a segment as not present.

2.10.1 Encoding rules notes

If a segment is to be continued across messages, use the extended encoding rules. These rules are defined in terms of the more general message continuation protocol (see Section 2.23.2, "Continuation messages and segments").

2.10.2 Version compatibility definition

The above rules for receiving HL7 messages and converting their contents to data values allow the following definition of a backward compatibility requirement between the 2.x versions of HL7:

a) New messages may be introduced.

b) New segments may be introduced to an existing message. In general these will be introduced at the end of a message, but they may be introduced elsewhere within the message if the segment hierarchy makes this necessary.

c) New fields may be added at the end of a segment, new components may be added at the end of a field, new subcomponents may be added at the end of a component, and a non-repeating field may be made repeating.

If a non-repeating field is made repeating, the first instance of that repeating field must have the same meaning as the non-repeating field had in the prior version of HL7.

For existing fields in existing segments, data types may be changed by the above rule (Section 2.10.2,c) if the leftmost (prior version) part of the field has the same meaning as it had in the prior version of HL7. In other words, if the new parts of the field (those that are part of the new data type) are ignored, what remains is the old field (defined by the old data type), which has the same meaning as it had in the prior version of HL7.

2.11 CHAPTER FORMATS FOR DEFINING HL7 MESSAGES

Subsequent chapters of this document describe messages that are exchanged among applications in functionally-specific situations. Each chapter is organized as follows:

a) **purpose**. This is an overview describing the purpose of the chapter, general information and concepts.

b) **trigger events and messages**. There is a list of the trigger events. For each trigger event the messages that are exchanged when the trigger event occurs are defined using the HL7 abstract message syntax as follows:

Each message is defined in special notation that lists the segment IDs in the order they would appear in the message. Braces, { . . . }, indicate one or more repetitions of the enclosed group of segments. (Of course, the group may contain only a single segment.) Brackets, [. . .], show that the enclosed group of segments is optional. If a group of segments is optional and may repeat it should be enclosed in brackets and braces, { [. . .] }.

Note:	[{...}] and {[...]} are equivalent.

Whenever braces or brackets enclose more than one segment ID a special stylistic convention is used to help the reader understand the hierarchy of repetition. For example, the first segment ID appears on the same line as the brace, two columns to the right. The subsequent segment IDs appear under the first. The closing brace appears on a line of its own in the same column as the opening brace. This convention is an optional convenience to the user. If there is conflict between its use and the braces that appear in a message schematic, the braces define the actual grouping of segments that is permitted.

c) **message segments**. The segments defined in a chapter are then listed in a functional order designed to maximize conceptual clarity.

d) **examples**. Complete messages are included.

e) **implementation considerations**. Special supplementary information is presented here. This includes issues that must be addressed in planning an implementation.

f) **outstanding issues**. Issues still under consideration or requiring consideration are listed here.

Consider the hypothetical triggering event **a widget report is requested**. It might be served by the Widget Request (WRQ) and Widget Report (WRP) messages. These would be defined in the Widget chapter (say Chapter XX). The Widget Request message might consist of the following segments: Message Header (MSH), Widget ID (WID). The Widget Report message might consist of the following segments: Message Header (MSH), Message acknowledgment (MSA), one or more Widget Description (WDN) Segments each of which is followed by a single Widget Portion segment (WPN) followed by zero or more Widget Portion Detail (WPD) segments.

2.11.1 HL7 abstract message syntax example

The schematic form for this hypothetical exchange of messages is shown in *Figure 2-5*:

Figure 2-5. Hypothetical schematic message

```
Trigger Event: WIDGET REPORT IS REQUESTED

WRQ             Widget Request           Chapter
MSH             Message Header             2
WID             Widget ID                  XX

WRP             Widget Report            Chapter
MSH             Message Header             2
MSA             Message acknowledgment     2
{ WDN           Widget Description         XX
  WPN           Widget Portion             XX
  { [WPD] }     Widget Portion Detail      XX
}
```

The WID, WDN, WPN, and WPD segments would be defined by the widget committee in the widget chapter, as designated by the Arabic numeral XX in the right column. The MSH and MSA segments, although included in the widget messages, are defined in another chapter. They are incorporated by reference into the widget chapter by the chapter number XX.

On the other hand, the widget committee might decide that the WPN and WPD segments should appear in pairs, but the pairs are optional and can repeat. Then the schematic for the WRP message would be as shown in *Figure 2-6*.

Figure 2-6. WPN and WPD segments in pairs

```
WRP             Widget Report            Chapter

MSH             Message Header             2
MSA             Message acknowledgment     XX
{ WDN           Widget Description         XX
  [ { WPN       Widget Portion             XX
      WPD       Widget Portion Detail      XX
   } ]
}
```

If the widget committee determined that at least one pair of WPN and WPD segments must follow a WDN, then the notation would be as shown in *Figure 2-7*.

Figure 2-7. At least one pair of WPN and WPD

```
WRP             Widget Report            Chapter

MSH             Message Header             2
MSA             Message acknowledgment     XX
{ WDN           Widget Description         XX
  {  WPN        Widget Portion             XX
     WPD        Widget Portion Detail      XX
  }
}
```

2.12 APPLICATION (LEVEL 7) PROCESSING RULES

2.12.1 Original and enhanced processing rules

The processing rules described here apply to all exchanges of messages, whether or not the HL7 encoding rules or Lower Layer Protocols are used. They represent the primary message processing mode. Certain variants are documented in Section 2.12.2, "APPLICATION (LEVEL 7) PROCESSING RULES." These include:

a) the application processing rules for a special processing mode, deferred processing. This mode remains in the specification only for backward compatibility.

b) an optional sequence number protocol

c) an optional protocol for continuing a very long message

The processing rules were extended in Version 2.2 of the Standard. The extensions provide a greater degree of flexibility in the way that messages can be acknowledged, as specified by several new fields in the Message Header segment. To provide backward compatibility with prior versions, the absence of these fields implies that the extended processing rules are not used. In the remainder of this section the extended mode is called the enhanced acknowledgment mode; the prior version is called the original acknowledgment mode.

Because the protocol describes an exchange of messages, it is described in terms of two entities, the initiating and responding systems. Each is both a sender and receiver of messages. The initiating system sends first and then receives, while the responding system receives and then sends.

In overview this exchange proceeds as follows:

Step 1 the initiating system constructs an HL7 message from application data and sends it to the responding system

Step 2 responder receives message and

2.1 when the original acknowledgment rules apply:

a) validates the message syntactically and against the detailed rules described in Section 2.12.1.1, "Initiation." If it fails, a reject message is constructed by the protocol software and returned to the initiator; if it does not fail, continue to the next step (2.1,b)

b) passes the message to the application, which:

1) creates a response message, or

2) creates an error message, or

3) creates a reject message

c) sends the response, error, or reject message

Initiator passes the message to the initiating application.

2.2 when enhanced acknowledgment rules apply:

a) the responding system receives the message and commits it to safe storage. This means that the responding system accepts the responsibility for the message in a manner that releases the sending system from any obligation to resend the message. The responding system now checks the message header record to determine whether or not the initiating system requires an accept acknowledgment message indicating successful receipt and secure storage of the message. If it does, the accept acknowledgment message is constructed and returned to the initiator.

b) at this point, the requirements of the applications involved in the interface determine whether or not more information needs to be exchanged. This exchange is referred to as an application acknowledgment and includes information ranging from simple validation to a complex application-dependent response. If the receiving system is expected to return application-dependent information, it initiates another exchange when this information is available. This time, the roles of initiator and responder are reversed.

The details follow.

2.12.1.1 Initiation

The initiating application creates a message with data values as defined in the appropriate chapter of this Standard. The fields shown below should be valued in the MSH segment (as defined under the MSH segment definition of this chapter). The message is encoded according to the applicable rules and sent to the lower level protocols, which will attempt to deliver it to the responding application. (For definitions of the MSH fields see Section 2.24.1, "MSH - message header segment.")

Field	Notes
MSH-3-sending application	
MSH-4-sending facility	
MSH-5-receiving application	
MSH-6-receiving facility	
MSH-7-date/time of message	This field is not used in the processing logic of the HL7 protocol. It is optional.
MSH-9-message type	
MSH-10-message control ID	Unique identifier used to relate the response to the initial message.
MSH-11-processing ID	
MSH-12-version ID	
MSH-13-sequence number	
MSH-14-continuation pointer	Used in implementation of message continuation protocol. See Sections 2.23.2, "Continuation messages and segments," 2.14.3, "Continuation of unsolicited display update message," and 2.15.4, "Interactive continuation or cancellation of response messages: original mode (display and record-oriented) and enhanced mode (display, tabular, and event replay)."

Certain other fields in the MSH segment are required for the operation of the HL7 encoding rules; they will not be relevant if other encoding rules are employed.

The event code in the second component of *MSH-9-message type* is redundantly shown elsewhere in some messages. For example, the same information is in the EVN segment of the ADT message. This is for compatibility with prior versions of the HL7 protocol. Newly-defined messages should only show the event code in *MSH-9-message type*.

2.12.1.2 Response

The protocol software in the responding system does one of the following:

2.12.1.2.1 When the original acknowledgment rules apply

Note:	Both *MSH-15-accept acknowledgment type* and *MSH-16-application acknowledgment type* are null or not present.

a) accepts the message

b) validates it against at least the following criteria:

1) the value in *MSH-9-message type* is one that is acceptable to the receiver

2) the value in *MSH-12-version ID* is acceptable to the receiver

3) the value in *MSH-11-processing ID* is appropriate for the application process handling the message

If any of these edits fail, the protocol software rejects the message. That is, it creates an ACK message with **AR** in *MSA-1-acknowledgment code*.

c) if the message passes the edits, the message is passed to the receiving application, which performs one of these functions:

1) process the message successfully, generating the functional response message with a value of **AA** in *MSA-1-acknowledgment code*.

- OR -

2) send an error response, providing error information in functional segments to be included in the response message with a value of **AE** in *MSA-1-acknowledgment code*.

- OR -

3) fail to process (reject) the message for reasons unrelated to its content or format (system down, internal error, etc.). For most such problems it is likely that the responding system will be able to accept the same message at a later time. The implementors must decide on an application-specific basis whether the message should be automatically sent again. The response message contains a value of **AR** in *MSA-1-acknowledgment code*.

d) passes the message to the initiating system

e) the protocol software in the initiating system passes the response message to the initiating application

In all the responses described above the following values are put in the MSA segment. Note that the field definitions for the MSA segment fields are in Section 2.24.2, “MSA - message acknowledgment segment”:

Field	**Notes**
MSA-1-acknowledgment code	As described above.
MSA-2-message control ID	*MSH-10-message control ID* from MSH segment of incoming message.

MSA-3-text message	Text description of error.
MSA-4-expected sequence number	As described in Section 2.23.1, "Sequence number protocol," (if the sequence number protocol is being used).
MSA-5-delayed acknowledgment type	For use only as described in Section 2.12.2, "Application (level 7) processing rules, deferred processing two phase reply (original acknowledgment mode only)."

The MSH segment in the response is constructed anew following the rules used to create the initial message described above. In particular, *MSH-7-date/time of message* and *MSH-10-message control ID* refer to the response message; they are not echoes of the fields in the initial message. *MSH-5-receiving application*, *MSH-6-receiving facility*, and *MSH-11-processing ID* contain codes that are copied from *MSH-3-sending application*, *MSH-4-sending facility* and *MSH-11-processing ID* in the initiating message.

2.12.1.2.2 When enhanced acknowledgment rules apply

Note:	At least one of *MSH-15-accept acknowledgment type* or *MSH-16-application acknowledgment type* is not null.

a) accepts the message

b) makes an initial determination as to whether or not the message can be accepted, based on factors such as:

1) the status of the interface

2) the availability of safe storage onto which the message can be saved

3) the syntactical correctness of the message, if the design of the receiving system includes this type of validation at this phase

4) the values of *MSH-9-message type*, *MSH-12-version ID*, and *MSH-11-processing ID*, if the design of the receiving system includes this type of validation at this phase

c) examines the Message Header segment (MSH) to determine whether or not the initiating system requires an accept acknowledgment.

If it does, the responding system returns a general acknowledgment message (ACK) with:

1) a commit accept (CA) in *MSA-1-acknowledgment code* if the message can be accepted for processing

2) a commit reject (CR) in *MSA-1-acknowledgment code* if the one of the values of *MSH-9-message type*, *MSH-12-version ID* or *MSH-11-processing ID* is not acceptable to the receiving application

3) a commit error (CE) in *MSA-1-acknowledgment code* if the message cannot be accepted for any other reason (e.g., sequence number error)

For this response, the following values are put in the MSA segment. Note that the field definitions for the MSA segment fields are in Section 2.24.2, "MSA - message acknowledgment segment"):

Field	Notes
MSA-2-message control ID	*MSH-10-message control ID* from the incoming message.
MSA-1-acknowledgment code	As described above.
MSA-3-text message	Text description of error.
MSA-4-expected sequence number	As described in Section 2.23.1, "Sequence Number Protocol" (if the sequence number protocol is being used).

The MSH segment in the response is constructed anew following the rules used to create the initial message described above. In particular, *MSH-7-date/time of message* and *MSH-10-message control ID* refer to the response message; they are not echoes of the fields in the initial message. *MSH-5-receiving application*, *MSH-6-receiving facility*, and *MSH-11-processing ID* contain codes that are copied from *MSH-3-sending application*, *MSH-4-sending facility* and *MSH-11-processing ID* in the initiating message.

Note: *MSH-15-accept acknowledgment type* and *MSH-16-application acknowledgment type* are not valued (not present or null). At this point, the accept portion of this message exchange is considered complete.

d) If the message header segment indicates that the initiating system also requires an application acknowledgment, this will be returned as the initial message of a later exchange.

For this response, the following values are put in the MSA segment. Note that the field definitions for the MSA segment fields are in Section 2.24.2, "MSA - message acknowledgment segment):

Field	Notes
MSA-2-message control ID	Identifies the initial message from the original initiating system as defined in Section 2.12.1.1, "Initiation."
MSA-1-acknowledgment code	Uses the application (processing) acknowledgment codes as described in Section 2.12.1.2.1, "When the original acknowledgment rules apply."
MSA-3-text message	Text description of error.

For this message, the receiving system acts as the initiator. Since the message it sends is application-specific, the layouts of these application-level response messages are defined in the relevant application-specific chapter. If needed, this application acknowledgment message can itself require (in *MSH-15-accept acknowledgment type*) an accept acknowledgment message (MSA). *MSH-16-application acknowledgment type*, however, is always null, since the protocol does not allow the application acknowledgment message to have an application acknowledgment.

At this point, the application acknowledgment portion of this message exchange is considered complete.

If the processing on the receiving system goes through multiple stages, chapter-defined messages may be used to relay status or informational changes to other systems (including the original initiating system). Such messages are not part of the acknowledgment scheme for the original message, but are considered to be independent messages triggered by events on the (original) responding system.

Note: The original acknowledgment protocol is equivalent to the enhanced acknowledgment protocol with *MSH-15-accept acknowledgment type* = NE and *MSH-16-application acknowledgment type* = AL, and with the application acknowledgment message defined so that it never requires an accept acknowledgment (*MSH-*

15-accept acknowledgment type = NE).

2.12.2 Application (level 7) processing rules, deferred processing two phase reply (original acknowledgment mode only)

(This section remains in the specification only for reasons of providing backward compatibility: it is to be used only with the original acknowledgment protocol. For the original acknowledgment protocol, it creates a generic form of an asynchronous application level acknowledgment, the MCF message.)

The application processing rules for deferred processing are described here. In this mode the responding system sends an acknowledgment to the initiating system that means the message has been placed in some type of secure environment (e.g., disk storage), and the receiving system commits to processing it within a reasonable amount of time, if a) the message contains the necessary information, and b) nothing causes the message's request for action to be canceled before the responding system processes the request.

Note: Neither of these two conditions is completely checked at the time of the first acknowledgment. They are both checked at the time of processing.

The receipt of the first delayed acknowledgment by the initiating system means that the responding system has taken responsibility for the subsequent processing of the message. This also implies that the initiating system no longer needs to keep the particular message in its current form to send out later. For example, if the sending system were maintaining a queue of outgoing messages, the particular message could be deleted from the output queue at this point.

The receipt of the second delayed acknowledgment message informs the initiating application of either: a) the application's successful processing of the initial message, or b) an error that prevented its processing. If the receiving application needs to return detailed change of status information, an application-specific message will be used. An example of the latter is the General Order message (ORM) described in Chapter 4.

The general delayed acknowledgment protocol is implemented on a site-specific and application-specific basis as needed. At a particular site, for a given transaction type the choices are:

a) do not allow deferred acknowledgments

b) all messages will have a deferred acknowledgment

c) only exceptional cases (errors) will receive the deferred acknowledgment

In overview the processing for options b) and c) proceeds as follows:

Initiator receives message from sending application and sends it to the responding system.

The responding system receives the message from the initiating system and

a) partially validates it syntactically and against the detailed rules described in Section 2.12.1, "Original and enhanced processing rules." This validation need not be complete but should be sufficient to determine the application that will ultimately respond to the message. If this validation fails, a reject message is constructed by the protocol software and returned to the initiator.

b) (if the message passes this validation) stores it and constructs a response message that simply acknowledges receipt. *MSA-5-delayed acknowledgment type* then has a value of **D**.

c) subsequently passes the message to the application, which:

1) creates a response message, or

2) creates an error message, or

3) creates a reject message

d) The protocol software sends the response, error, or reject message to the initiating system as an unsolicited update with no value present in *MSA-5-delayed acknowledgment type*.

The protocol software of the initiating system responds to the response, error, or reject message with simple acknowledgment and passes it to the initiating application.

The details follow.

2.12.2.1 Initiation

The rules for creating the initial message are exactly as defined in Section 2.12.1, "Original and enhanced processing rules," for the original acknowledgment rules.

2.12.2.2 Response

The processing in the responding system follows this pattern:

a) the protocol software accepts the message and validates it against at least the following criteria:

1) the value in *MSH-9-message type* is one that is acceptable to the receiver

2) the value in *MSH-12-version ID* is acceptable to the receiver

3) the value in *MSH-11-processing ID* is appropriate for the application process handling the message

If any of these edits fail, the protocol software rejects the message. That is, it creates an ACK message with **AR** in *MSA-1-acknowledgment code*.

b) If the message passes the edits, the protocol software stores it and a generates a response message of type ACK with a value of **AA** in *MSA-1-acknowledgment code* and **D** in *MSA-5-delayed acknowledgment type*.

c) Subsequently the protocol software passes the message to the application, which performs one of these functions:

1) processes the message successfully, generating the functional response message (message type MCF) with a value of **AA** in *MSA-1-acknowledgment code*.

- OR -

2) creates an error response, providing error information in functional segments to be included in the response message, which has a value of **AE** in *MSA-1-acknowledgment code*.

- OR -

3) fails to process (rejects) the message for reasons unrelated to its content or format (system down, internal error, etc.) For most such problems it is likely that the responding system will be able to accept the same message at a later time. The implementors must decide on an application-specific basis whether the message should be automatically sent again. The MSA segment of the response message contains a value of **AR** in *MSA-1-acknowledgment code*.

d) the application passes the message to the protocol software, which constructs a message of type MCF with **F** in *MSA-5-delayed acknowledgment type*.

e) the protocol software passes the message to the initiating system as an unsolicited update.

f) the protocol software in the initiating system passes the response message to the initiating application and generates a simple ACK message. No value is present in *MSA-5-delayed acknowledgment type*.

All other values are put in the MSA segment as described in Section 2.12.1, "Original and enhanced processing rules."

2.13 ACKNOWLEDGEMENT MESSAGES

Acknowledgment messages may be defined on an application basis. However the simple general acknowledgment message (ACK) may be used where the application does not define a special message (application level acknowledgment) and in other cases as described in Section 2.12.1, "Original and enhanced processing rules." ***The MCF message is included only for backward compatibility with HL7 Version 2.1 (see Section 2.12.2, "Application (level 7) processing rules, deferred processing two phase reply (original acknowledgment mode only)").***

2.13.1 ACK - general acknowledgment

The simple general acknowledgment (ACK) can be used where the application does not define a special application level acknowledgment message or where there has been an error that precludes application processing. It is also used for accept level acknowledgments. The details are described in Section 2.12.1, "Original and enhanced processing rules."

ACK	General acknowledgment	Chapter
MSH	Message Header	2
MSA	Message acknowledgment	2
[ERR]	Error	2

2.13.2 MCF - delayed acknowledgment

This message remains in the specification only for reasons of backward compatibility with HL7 Version 2.1. It is used as part of the protocol which creates a generic form of an asynchronous application level acknowledgment, the MCF message. See Section 2.12.1.2, "Response."

The first MCF message, sent after the initial receipt has the following structure.

MCF	Delayed Acknowledgment	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2

The second MCF message, sent after application processing, has this structure:

MCF	Delayed Acknowledgment	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2

2.14 DISPLAY MESSAGES

2.14.1 Display vs. record-oriented messages

HL7 messages may contain:

a) data in a format suitable for display purposes (display-oriented data), or

b) data in a format which explicitly denotes field content (record-oriented)

This record-oriented data appears in two forms: tabular (defined in Section 2.15, "QUERIES," of this chapter); and application-message segment-oriented (defined in the various application sections of this document, see Chapters 3-12).

A display message can be generated to fit a variety of needs for unsolicited updates between systems. These are situations where the update information does not need to be captured by the receiving system's database, but only displayed, either on a visual medium (such as a PC, workstation or a CRT) or on printed medium.

The Unsolicited Display Message describes the display-oriented message. It is the unsolicited version of the generalized Response display message (see Section 2.15, "QUERIES"). It is acknowledged by a general acknowledgment message (ACK).

The content and format of record-oriented messages require functionally-specific capabilities. The technical committees responsible for functionally-specific chapters define them within those chapters.

2.14.2 UDM/ACK - unsolicited display update message (event Q05)

There is a simple HL7 message that allows for unsolicited display update messages to be sent in HL7 format from one system to another.

Trigger events for the unsolicited update are generally the completion of a particular action (concerning a given patient). For example, a lab test might be completed, generating a STAT unsolicited display message to be sent to the appropriate location.

UDM	Unsolicited Display Message	Chapter
MSH	Message Header	2
URD	Results/Update Definition	2
[URS]	Results/Update Selection Criteria	2
{ DSP }	Display Data	2
[DSC]	Continuation Pointer	2

ACK	General Acknowledgment	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2

2.14.3 Continuation of unsolicited display update message

Like other types of HL7 messages, the UDM message can be continued by use of the DSC segment and *MSH-14-continuation pointer*. Thus if a UDM needs to be continued as three separate UDM messages, the first message would contain:

```
MSH          (no continuation pointer)
URD
[URS]
{  DSP  }
DSC          (with continuation pointer)
```

The second message would contain:

```
MSH          (continuation pointer (to first message))
{  DSP  }
DSC          (with continuation pointer)
```

The last message would then contain:

```
MSH          (continuation pointer (to second message))
{  DSP  }
             (no DSC, since last)
```

Note: This scheme works equally well with non-display messages, such as the Unsolicited Update ORU message (see Chapter 7).

Since these are unsolicited messages, intervening messages (from other systems) may be sent to the receiving application while the sections of the particular message are being continued. *MSH-14-continuation pointer* enables the receiving system to keep track of extraneous intervening messages.

2.15 QUERIES

2.15.1 Display vs. record-oriented queries (original mode, embedded query language, virtual table, and stored procedure queries); and event replay requests

This section addresses the framework defined for responses to queries formatted as display or record-oriented (original mode, tabular, and event replay response messages). The detailed content of the record-oriented query and response messages is defined by the technical committees responsible for the functionally-specific chapters: their basic forms are defined in this chapter.

2.15.2 Message definition

This section defines the following message:

- a generalized query, compatible with previous releases of HL7 (henceforth referred to as the original query)
- An Embedded Query Language query, which supports free-form select statements, based on the query language of choice (e.g., SQL)
- a virtual table request query which supports queries against server database tables (virtual or actual) based on specific selection criteria
- a stored procedure request, which enables an application on one system to execute a stored procedure on another system, which is coded to extract specific data
- a generalized display response message in which the responding system formats the data for direct output to a display device (in both original and enhanced modes)
- an event replay request message, which is used to request data formatted as an event replay response
- a tabular response message in which the responding system formats the data in a relational format, as rows and columns
- an event replay response, in which the responding system formats the data on the basis of an application-specific segment-oriented (record-oriented) message

Note :	The response paradigm and basic form of the original mode record-oriented query are defined in this chapter, while the detailed forms are defined in the functionally-specific chapters.

The following represents typical examples of queries supported by the Standard:

a) for data regarding a single patient, e.g., send all lab results for patient #123456

b) for data regarding multiple patients, e.g., send the list of patients whose attending physician is Dr. #123

c) for data that is not patient related, e.g., send the age specific normal values for serum protein

The variety of potential queries is almost unlimited. There was no attempt here to define a Standard that would cover every possible query. Instead, the Standard embraces the most common queries that are likely to occur in a hospital. For each common query there is a corresponding unsolicited update. See Section 2.14.2, "UDM/ACK - unsolicited display update message (event Q05)."

In particular, there is no implication that a specific system must support generalized queries to comply with the Standard. Rather, these transactions provide a format, or a set of tools to support queries to the extent desired by the institution. The resources available and local policies will influence the type of queries that are implemented.

2.15.3 Immediate vs. deferred response

Responses to queries can be either immediate or deferred. The query describes this as the expected response time. In the immediate mode, the responding process gives the response immediately or in a short period during which the requesting process will wait for the response.

2.15.4 Interactive continuation or cancellation of response messages: original mode (display and record-oriented) and enhanced mode (display, tabular, and event replay)

One use of queries is to retrieve data from one application for presentation to users of another. This approach might be used for users of a patient care system retrieving data from lab or other ancillaries. It also might permit users of a pharmacy system to retrieve a patient's lab results from the lab system or non-pharmacy order data from the patient care system. Almost any other application system could be the source of data or the system initiating the query for its users.

Of particular interest is the case where the inquiring user formulates the query online at the terminal of one system and waits while that system sends the query to another. He gets the response and displays it at their terminal. When the user is formulating such a query she may only have limited understanding of what data is available for a given patient. Sometimes the user's preference would be to make a simple query such as give me recent data in reverse chronological sequence rather than give me data for yesterday, since there may be some data available for today, or there may be data from two days ago that is of interest. The user will look at the data returned and simply quit looking at it after he or she has found the part that is of interest. (The time frames or the sort sequence may differ, or the user may wish to impose some selectivity on the response, but the general principle remains the same. The user would prefer to make a vague statement of the interest, have the data presented in order of decreasing likelihood of interest, and quit when he or she has seen enough.)

While beneficial to the user, this way of requesting data could be very burdensome when the resulting query takes place over an inter-application interface. If the responding system were to retrieve, format, and send all the data the user might like to see, the processing load would be extremely high and the response time unacceptable.

The continuation query provides a way to permit the users to formulate queries loosely while limiting the processing burden on the responding system. The initial query specifies the general constraints of the query and an amount of data to be returned. (For example, the query might be for lab results for patient #12379 and 44 lines would be requested.) The responding system retrieves and formats the specified amount of data and returns it with a special key field, *DSC-1-continuation pointer*. The initiating system presents the requested data to the user and retains the continuation pointer field for use if another query is needed. The internal structure of the value is not known to the initiating system.

If, after viewing the data, the user requests more, the initiating system sends the query again in a format that is identical with the first, except that the *DSC-1-continuation pointer* value is included and the requested amount of data may be changed. The responding system uses the continuation pointer field as a key into its database to continue retrieval and formatting of the results. If the user does not request more data, no further messages are exchanged.

The initiating system may also explicitly terminate the query by sending a "cancel query" message. This message is formatted just like a continuation query, except that the event-type (*MSH-9-message type*) is set equal to CNQ. Although the effect to the initiating system is the same as if it had not sent any message (no further query data is received), receipt of this message by the responding system enables it to discard any unsent continuation data that might be queued.

2.15.5 Logical display break points

Often the lines of display text will fall into logical groups that differ from the physical size of screen or printer page. For example, a complete battery or an entire radiology report might be thought of as comprising a logical group, though they might have as few as six or as many as 120 lines. Knowledge of the logical break points in the display data can be useful to the application system that is displaying or printing data. For this reason, *DSP-4-logical break point* is used. The sending application (the one that formats the data) places the logical break points where appropriate. If there is a particular ancillary result ID associated with the data delineated by *DSP-4-logical break point*, the value of this ID also can be returned in *DSP-5-result ID*. Then if the user selects the area of the display delineated by *DSP-4-logical break point*, the displaying system can query for the associated *DSP-5-result ID*.

2.16 QUERY TRIGGER EVENTS AND MESSAGE DEFINITIONS

These are the trigger event types associated with queries:

a) a need occurs for immediate access to data that may be available from another application, this may be an initial request for data or a continuation

b) a need occurs for deferred access to data that may be available from another application

Where an original-style query is involved, these trigger events are served by the query message (QRY). When display data is involved, these trigger events are served by the Query (QRY) and Display Response (DSR) and General Acknowledgment (ACK) messages. When the query is for record-oriented data, the QRY message is used, but the response message is specific to a functional area. Record-oriented queries are described in detail in the relevant application chapters. Display-oriented response are described in detail in this chapter.

Although the query message of an event replay message is described in detail in this chapter, event replay response messages will be described in detail in the relevant application chapters.

Where an Embedded Query Language query is involved, these trigger events are served by the Embedded Query Language query (EQQ).

Where a virtual table query is involved, these trigger events are served by the Virtual Table Query (VQQ).

Where a stored procedure request is involved, these trigger events are served by the Stored Procedure request (SPQ) message.

For each of these query messages, one of the following response messages is used.

When display data is involved, these queries are responded to with the Display Response (DSR) message.

When tabular data is involved, these queries are responded to with the Tabular Data Response (TBR).

Where an event replay query is involved, these trigger events are served by the Event Replay Query (ERQ) and the Event Replay Response (RQQ) messages.

Where an original QRY query message is used to request record-oriented data, the response message is specific to a functional area. Application-segment record-oriented queries are described in other chapters (3-8).

Display-oriented, tabular response messages, and event replay response messages are described here.

Each trigger event is listed below, with the applicable form of message exchange.

2.17 ORIGINAL MODE QUERIES

2.17.1 QRY/DSR - original mode display query - immediate response (event Q01)

QRY	Query Message	Chapter
MSH	Message Header	2
QRD	Query Definition	2
[QRF]	Query Filter	2
[DSC]	Continuation Pointer	2

DSR	Display Response Message	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2
[QAK]	Query Acknowledgment	2
QRD	Query Definition	2
[QRF]	Query Filter	2
{ DSP }	Display Data	2
[DSC]	Continuation Pointer	2

The QRF and QRD segments from the QRY are echoed back in the response. The DSC segment contains the continuation pointer, if it is not null (*DSC-1-continuation pointer*).

2.17.1.1 Original mode display query variants

If a display query has more than a single type of response (i.e., a DSR message with a different meaning, requiring different processing on the querying system), the second component of the Message Type field of the MSH segment may be used to indicate the response event type. For example, an ancillary name search display query may be defined using the query event code of DNM. The display response to such a query may be either a list of name matches (response event type is DNM) or the ancillary's display results for an

exact match to the name query (response event type is NRS). See *HL7 table 0003 - Event type code* and field notes for *MSH-9-message type*.

2.18 ORIGINAL MODE DEFERRED ACCESS

For clarity, A is the system initiating the query and B is the system sending the responses. Multiple queries and responses are permitted within single messages. The responses to a given query may be broken into several separate DSR messages. A single DSR message may contain responses to more than one QRY.

2.18.1 QRY/QCK - deferred query (event Q02)

For clarity, A is the system initiating the query and B is the system sending the responses. Multiple queries and responses are permitted within single messages. The responses to a given query may be broken into several separate DSR messages. A single DSR message may contain responses to more than one QRY.

QRY (A to B)	Query Message	Chapter
MSH	Message Header	2
QRD	Query Definition	2
[QRF]	Query Filter	2
[DSC]	Continuation Pointer	2

QCK (B to A)	Query General Acknowledgment	Chapter
MSH	Message Header	2
MSA	Message acknowledgment	2
[ERR]	Error	2
[QAK]	Query Acknowledgment	2

2.18.2 DSR/ACK - deferred response to a query (event Q03)

Later, perhaps more than once.

DSR (B to A)	Display Response Message	Chapter
MSH	Message Header	2
[MSA]	Message Acknowledgment	2
[ERR]	Error	2
[QAK]	Query Acknowledgment	2
QRD	Query Definition	2
[QRF]	Query Filter	2
{ DSP }	Display Data	
[DSC]	Continuation Pointer	2

ACK (A to B)	General Acknowledgment	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2

Note: All record-oriented original mode and all enhanced mode queries follow the immediate and deferred acknowledgment modes defined in Section 2.18.1, “QRY/QCK - deferred query (event Q02).”

2.19 ENHANCED MODE QUERY MESSAGES

2.19.1 EQQ - embedded query language query (event Q01)

EQQ	Embedded Query Language Query	Chapter
MSH	Message Header	2
EQL	EQL Definition	2
[DSC]	Continuation Pointer	2

2.19.2 VQQ - virtual table query (event Q01)

VQQ	Virtual Table Query	Chapter
MSH	Message Header	2
VTQ	VTQ Definition	2
[RDF]	Table Row Definition	2
[DSC]	Continuation Pointer	2

2.19.3 SPQ - stored procedure request (event Q01)

SPQ	Stored Procedure Request	Chapter
MSH	Message Header	2
SPR	Store Procedure Request	2
[RDF]	Table Row Definition	2
[DSC]	Continuation Pointer	2

2.19.4 RQQ - event replay query (event Q01)

RQQ	Event Replay Query	Chapter
MSH	Message Header	2
ERQ	Event Replay Query	2
[DSC]	Continuation Pointer	2

2.20 ENHANCED QUERY MODE RESPONSE MESSAGES

2.20.1 EDR - enhanced display response

EDR	Enhanced Display Response	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2
QAK	Query Acknowledgment	2
{ DSP }	Display Data	2
[DSC]	Continuation Pointer	2

2.20.2 TBR - tabular data response

TBR	Tabular Data Response	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2
QAK	Query Acknowledgment	2
RDF	Table Row Definition	2
{ RDT }	Table Row Data	2
[DSC]	Continuation Pointer	2

2.20.3 ERP - event replay response

ERP	Event Replay Response	Chapter
MSH	Message Header	2
MSA	Message Acknowledgment	2
[ERR]	Error	2
QAK	Query Acknowledgment	2
ERQ	Event Replay Query	2
...		
...		
[DSC]	Continuation Pointer	2

Note: The remainder of this message is defined by the contents of the corresponding segment-oriented record-oriented unsolicited update message, excluding the MSH, as defined by the function-specific chapters of this specification. The input parameter list may be satisfied by more than one record-oriented unsolicited update message: in this case, the segment group after the ERQ segment may repeat.

Note: When this message is continued, the continuation messages should include the MSH, MSA, [ERR], QAK, ERQ, and [DSC] segments, as well as the segments indicated by the ellipsis (...) in the response definition and the DSC should be used only at the end of the group corresponding to the record-oriented unsolicited update message.

Enhanced mode record-oriented response messages note: The RDF segment from the EQQ, VQQ and SPQ messages, and the ERQ segment from the EQQ message, are echoed back in their respective responses. The DSC segment contains the continuation pointer, if it is not null (*DSC-1-continuation pointer*).

2.21 QUERY MESSAGE IMPLEMENTATION CONSIDERATIONS

2.21.1 Original mode implementation considerations

a) The particular allowable values for the filters in the QRD and QRF segments are determined among cooperating applications during implementation.

b) The format chosen for the query segments is very general. This might be read by prospective implementors to imply that the requirement for using the Standard is the ability to respond to a wide variety of inquiries. This is not the intent. The format here can be used with specific restrictions in any interface.

2.21.2 Enhanced mode implementation considerations: definition of tables and "virtual tables"

a) The particular allowable values for the EQL, VTQ, SPR, and RDF segments are determined among cooperating applications during implementation.

b) The formats chosen for the query messages are very general. This might be read by prospective implementors to imply that the requirement for using the Standard is the ability to respond to a wide variety of inquiries. This is not the intent. The format here can be used with specific restrictions in any interface.

c) The contents of the tables expressed as TBR response messages are defined by the functional chapters. Where an existing HL7 segment contains the fields needed to form a row of a tabular response, the segment ID can be referenced. For example, if a table of patients is needed, where each row represents a patient and each column a field from the PID segment, then the PID can be referenced as a table, also sometimes referred to as a "virtual table."

Where each row is comprised of fields from multiple HL7 segments, the functional chapters may define additional tables. For example, a table may be defined to respond to insurance queries where each row represents a patient, and is comprised of columns derived from the PID segment and the insurance segments *(IN1-IN4)*.

2.22 QUERY ERROR RESPONSE

If an application detects an error while processing a query, it responds by returning an Application Error (AE) or Application Reject (AR) condition in the MSA segment of the applicable query response message (DSR, TBR or ERP). The responding application values *MSA-6 error condition* with the appropriate error code and message. The continuation (DSC) segment is not sent or, if it is, its continuation pointer field (*DSC-1-continuation pointer*) is null. If the QAK segment is being used, the field *QAK-2-query response status* is valued appropriately with either AE (application error) or AR (application reject).

Note: If the responding application successfully processes the query, but is unable to find any qualifying data, this is not an error condition. The responding application returns an Application Accept (AA) in the MSA segment of the query response message, but does not return any data segments (DSP, RDT, or event replay segments). The continuation (DSC) segment is not sent or, if it is, its continuation pointer field (*DSC-1- continuation pointer)* is null. If the QAK segment is being used, the field *QAK-2-query response status* is valued with NF (no data found, no errors).

2.23 SPECIAL HL7 PROTOCOLS

This section contains several extensions to the basic HL7 message protocol. These extensions represent implementation choices, and are to be used on a site-specific and application-specific basis as needed.

2.23.1 Sequence number protocol

For certain types of data transactions between systems the issue of keeping databases synchronized is critical. An example is an ancillary system such as lab, which needs to know the locations of all inpatients to route stat results correctly. If the lab receives an ADT transaction out of sequence, the census/location information may be incorrect. Although it is true that a simple one-to-one acknowledgment scheme can prevent out-of-sequence transactions between any two systems, only the use of sequence numbers can prevent duplicate transactions.

Note: Although this sequence number protocol is limited to the use of sequence numbers on a single transaction stream between two applications, this sequencing protocol is sufficiently robust to allow the design of HL7-compatible store-and-forward applications.

a) definitions, initial conditions:

   1) the system receiving the data stream is expected to store the sequence number of the most recently accepted transaction in a secure fashion before acknowledging that transaction. This stored sequence number allows comparison with the next transaction's sequence number, and the implementation of fault-tolerant restart capabilities.

   2) the initiating system keeps a queue of outgoing transactions indexed by the sequence number. The length of this queue must be negotiated as part of the design process for a given link. The minimum length for this queue is one.

   3) the sequence number is a positive (non-zero) integer; and it is incremented by one (by the initiating system) for each successive transaction.

b) starting the link:

   1) the value of 0 (zero) for a sequence number is reserved: it is allowed only when the initiating system (re-)starts the link.

   2) if the receiving system gets a transaction with a 0 (zero) in the sequence number field, it should respond with a general acknowledgment message whose MSA contains a sequence number one greater than the sequence number of the last transaction it accepted in the Expected Sequence Number field. If this value does not exist (as on the first startup of a given link), the MSA should contain a sequence number of -1, meaning that the receiving system will use the positive, non-zero sequence number of the first transaction it accepts as its initial sequence number (see resynching the link, item **e** below).

3) the initiating system then sends the transaction indexed by the expected sequence number (if that expected transaction is still on its queue). Otherwise the link is frozen until an operator intervenes.

c) normal operation of the link:

As it accepts each transaction, the receiving system securely stores the sequence number (which agrees with its expected sequence number), and then acknowledges the message by echoing the sequence number in *MSA-4-expected sequence number*.

d) error conditions (from point of view of initiating system). These are generated by the receiving system, by its comparison of the sequence number sent out (with the MSH in *MSH-13-sequence number*) with the expected sequence number (*MSA-4-expected sequence number* received with the MSA).

1) **expected sequence number is one greater than current value**. The previous acknowledgment was lost. That transaction was sent again. Correct by sending next transaction.

2) **expected sequence number less than current value**. Initiating system can try starting again by issuing a transaction with a sequence number of zero; or freeze the link for operator intervention.

3) **other errors**: freeze the link for operator intervention

e) forcing resynchronization of sequence numbers across the link. The value of -1 for a sequence number is reserved: it is allowed only when the initiating system is resynching the link. Thus if the receiving system gets a value of -1 in the sequence number field, it should return a general acknowledgment message with a -1 in the expected sequence number field. The receiving system then resets its sequence number, using the non-zero positive sequence number of the next transaction it accepts.

f) notes

When the initiating system sends a message with a sequence number of **0** or **-1** (see b or e above), the segments beyond the MSH need not be present in the message, or, if present, all fields can be null. In terms of the responding system, for these two cases, only a General acknowledgment message is needed.

2.23.2 Continuation messages and segments

See Section 2.15.4, "Interactive continuation or cancellation of response messages: original mode (display and record-oriented) and enhanced mode (display, tabular, and event replay)," for a discussion of the continuation pointer segment and the continuation pointer field, and their use in the continuation of responses to queries and in the continuation of unsolicited update messages.

Besides the need to continue a message, there are occasional implementation conditions that force the continuation of a segment across messages. Such continued segments require the use of the ADD segment as follows:

a) the segment being continued (call it ANY for this example) is ended at an arbitrary character position and terminated with the standard segment terminator (carriage return).

b) the following segment is the ADD segment. When it follows a segment being continued, the ADD segment contains no fields. Whether the message being continued is a response to a query, or an unsolicited update, the receiving system will use the continuation pointer (with the ADD segment) to continue the message.

c) when a response (to a query) is continued, the first segment after the QRD and QRF (on a continued query) will be the ADD segment. All the fields after the ADD segment identifier (fields 1-N) will be the continuation of the ANY segment. The receiving system will then use the continuation pointer to join the two parts of the ANY segment and continue processing the message.

d) for the continuation of an unsolicited update message, the ADD segment will be the first segment after the MSH segment. The receiving system will use the continuation pointer field in the MSH segment to identify the message being continued, and then use the ADD segment as described in c) to join the two parts of the ANY segment.

e) limitations: MSH, MSA, DSC, PID, QRD, QRF, URD and URS segments cannot be continued.

f) although the UU example given below is for a display message, there is nothing in the protocol to prevent a record-oriented UU from being continued in this fashion. In the unsolicited display message, the ADD record on the continuation comes just after the URD/[URS] pair instead of directly after the MSH.

g) transaction flow for a continued query-response pair with an ADD segment:

1) first query and response:

```
MSH
QRD
[QRF]

MSH
MSA
[ ERR ]
QRD
[QRF]
    {  DSP  }    (last DSP segment is incomplete)
ADD              (ADD segment contains no fields)
DSC
```

2) second query and response:

```
MSH

QRD

[QRF]

DSC          (contains the continuation pointer from the DSC segment of
prior response

           message)



MSH

MSA

[ ERR ]

QRD

[QRF]

ADD          (contains the remainder of the last DSP segment of the
           previous response)

   {DSP}  (remaining DSP segments are complete)
```

Note:	This second response could itself be continued with a second DSC and (if needed) a second ADD segment prior to it. This paradigm also applies to both original mode and enhanced mode queries of display and record-oriented types.

f) transaction flow for a continued unsolicited message with a continued segment.

1) first unsolicited message and acknowledgment:

```
MSH

URD

[ URS ]

   {DSP}     (last DSP is incomplete)

ADD              (contains no fields)

DSC              (Continuation segment)



MSH          (General acknowledgment)

MSA

[ ERR ]
```

2) second unsolicited message and acknowledgment:

```
MSH          (contains continuation pointer from DSC segment
          of prior message)

ADD          (contains remainder of data from continued DSP
          segment from prior message)

   {DSP}
```

Note:	This second message could itself be continued with a second DSC and (if needed) a second ADD segment prior to it.

```
MSH          (General acknowledgment)

MSA

[ ERR ]
```

2.23.3 HL7 batch protocol

There are instances when it is convenient to transfer a batch of HL7 messages. Common examples would be a batch of financial posting detail transactions (DFT's) sent from an ancillary to a financial system. Such a batch could be sent online using a common file transfer protocol, or offline via tape or diskette.

2.23.3.1 HL7 batch file structure

The structure of an HL7 batch file is given by the following (using the HL7 abstract message syntax)

```
[FHS]               (file header segment)

{ [BHS]             (batch header segment)

 { [MSH             (zero or more HL7 messages)

   ....

   ....

   ....

] }

  [BTS]              (batch trailer segment)

}

[FTS]                (file trailer segment)
```

Notes:

The sequence numbering protocol has a natural application in batch transfers. See the discussion of batch acknowledgments that follows.

Although a batch will usually consist of a single type of message, there is nothing in the definition that restricts a batch to only one message type.

The HL7 file and batch header and trailer segments are defined in exactly the same manner as the HL7 message segments. Hence the HL7 message construction rules of Section 2.10, "MESSAGE CONSTRUCTION RULES," can be used to encode and decode HL7 batch files.

There are only two cases in which an HL7 batch file may contain zero HL7 messages:

a) a batch containing zero HL7 messages may be sent to meet a requirement for periodic submission of batches when there are no messages to send.

b) a batch containing zero negative acknowledgment messages may be sent to indicate that all the HL7 messages contained in the batch being acknowledged are implicitly acknowledged. See Section 2.23.3.3, "Related segments and data usage."

2.23.3.2 Related segments and data usage

The following segments defined in Section 2.24 , "MESSAGE CONTROL SEGMENTS," relate to the HL7 Batch Protocol:

BHS	Batch Header
BTS	Batch Trailer
FHS	File Header
FTS	File Trailer

The BTS segment contains a field, *BTS-3-batch totals*, which may have one or more totals drawn from fields within the individual messages. The method for computing such totals will be determined on a site or application basis unless explicitly stated in a functional chapter.

2.23.3.3 Acknowledging batches

In general, the utility of sending batches of data is that the data is accepted all at once, with errors processed on an exception basis. However, it is a permissible application of HL7 to acknowledge all messages. Several options for acknowledgment are given and will be chosen on an application basis. In these cases, the sequence numbering protocol can be useful to the applications processing the batch.

The options are:

a) all messages are acknowledged in the response batch.

b) the receiving system prints some form of batch control report, which is then dealt with manually by personnel at the sending system. No acknowledgments are performed by the protocol software.

c) an automated acknowledgment batch is created containing acknowledgment messages only for those messages containing errors. In this mode an empty acknowledgment batch may be created (i.e., an HL7 batch file without any HL7 acknowledgment messages).

In each case where there is a response batch, its format is a batch of individual messages. Each individual message is in the format defined for an online response in the chapters. Consider, for example, a batch that might be constructed to respond to a batch of Detailed Financial Transactions (Chapter 6). The messages in the response batch would consist entirely of ACK messages, since ACK is the response shown in Chapter 6.

When batches are retransmitted after the correction of errors, *BHS-12-reference batch control ID* should contain the batch control ID of the original batch.

2.23.3.4 Batch message as a query response

The HL7 query also can be used to query for a batch in the following manner:

a) use the value BB or BL of *QRD-5-deferred response type* to specify a batch response. The query will be acknowledged with a general acknowledgment as in the Deferred Access example above (see Section 2.18.2, "QRY/QCK - deferred query (event Q02)").

b) in addition, insert into the batch file the QRD and QRF segments as follows:

```
[FHS]                  (file header segment)
 { [BHS]               (batch header segment)
   [QRD]               (the QRD and QRF define the
   [QRF]                query that this batch is a response to)


   { MSH               (one or more HL7 messages)
      ....
      ....
      ....
   }


   [BTS]              (batch trailer segment)
}
[FTS]                 (file trailer segment)
```

c) the acknowledgment of a batch is described in this chapter (see Section 2.23.3.3, "HL7 batch protocol").

2.23.4 Modes for updating via repeating segments

When groups of repeating segments appear within a message it is not obvious from the basic HL7 abstract message syntax how best to apply the new group of repeating segments on the receiving system. HL7 suggests two methods: the "snapshot" mode and the "action code/unique identifier" mode.

Background:

The segments which repeat in HL7 messages Patient Administration (ADT)/Financial Information messages (AL1, DG1, PR1, GT1, IN1, IN2, IN3, NK1 , NTE) present a problem if the requirement is to update only part of the information previously sent. Prior to Version 2.3 of the Standard, all such repeating segments had to be sent again in the update, because there was no way to indicate which ones changed and which ones did not. For example, if two DG1 segments were sent originally (containing a primary and secondary diagnosis), and then if a tertiary diagnoses needed to be sent, the sending system had to send all diagnoses which were currently valid, that is, three DG1 segments (containing primary, secondary and tertiary diagnosis). This

way of doing things is referred to as the "snapshot" mode. In this mode, everything (all repeating segments) must be sent with every subsequent message in the series of messages.

In the Order Entry, Observation Reporting, and Master Files chapters, action codes (e.g., order control codes and result status codes) and unique identifiers (e.g., placer and filler numbers) are currently specified as part of the ORC, OBR, OBX and MFE segments. So, except for the NTE segments, this problem exists mainly for the Patient Administration and Financial Management chapter segments.

For systems implementing Version 2.3 or higher, if a particular repeating segment can be updated by either of these two modes, the parties concerned will determine by agreement on a site-specific basis whether an interface will use the "snapshot" mode or the "action code/unique identifier" mode.

2.23.4.1 Snapshot mode update definition

In the "snapshot" mode, the group of repeating segments from the incoming message replaces the prior group of repeating segments on the receiving system. This is equivalent to a deletion of the prior group followed by the addition of the new group. The snapshot mode is the usual mode in Version 2.2 and 2.1 implementations of HL7, but it is also available for Version 2.3 and future versions. To specify "delete all of the segments in this repeating group" in the snapshot mode, send a single segment with "delete data" indicated for all fields.

For example, if the following DG1 segment is in an ADT update message (for an inpatient stay):

```
DG1|""|""|””|””|””|""|””|””|””|””|””|””|””|””|””|””|””<cr>
```

and the snapshot mode is being used, this indicates that all previously-transmitted diagnoses for this inpatient stay should be deleted.

2.23.4.2 Action code/unique identifier mode update definition

In the "action code/unique identifier" mode, each member of a repeating group of segments must have a unique identifier (equivalent to the filler number in observational reports messages). The choice of delete/update/insert is determined by the action code (equivalent to the result status in observational reports messages). Refer to *HL7 table 0206 - Segment action code* for valid values.

Table 0206 - Segment action code

Value	Description
A	Add/Insert
D	Delete
U	Update

The *unique identifier* is defined in a general manner as follows: it uniquely identifies one of multiple repetitions of the primary entity defined by the repeating segment in a way that does not change over time. It is not dependent on any particular message identifier level (MSH) fields; it functions across messages, not just within a message. The *unique identifier* will be chosen on a segment-specific basis, depending on the primary entity referenced by the segment. For some cases, such as a diagnosis code, it may be a CE data type. For others, such as a person identifier, it may be a CX data type. For others it may be an EI (entity identifier) data type.

Note: This mode is available for use only for new segments for Version 2.3 and for new segments in future versions

2.24 MESSAGE CONTROL SEGMENTS

The following segments are necessary to support the functionality described in this chapter.

> **Note:** The HL7 message construction rules define the standard HL7 encoding rules, creating variable length delimited messages from the segments defined below. Although only one set of encoding rules is defined as a standard in HL7 Version 2.3, other encoding rules are possible (but since they are non-standard, they may only used by a site-specific agreement).

2.24.1 MSH - message header segment

The MSH segment defines the intent, source, destination, and some specifics of the syntax of a message.

Figure 2-8. MSH attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	1	ST	R			00001	Field Separator
2	4	ST	R			00002	Encoding Characters
3	180	HD	O			00003	Sending Application
4	180	HD	O			00004	Sending Facility
5	180	HD	O			00005	Receiving Application
6	180	HD	O			00006	Receiving Facility
7	26	TS	O			00007	Date/Time Of Message
8	40	ST	O			00008	Security
9	7	CM	R			00009	Message Type
10	20	ST	R			00010	Message Control ID
11	3	PT	R			00011	Processing ID
12	8	ID	R		0104	00012	Version ID
13	15	NM	O			00013	Sequence Number
14	180	ST	O			00014	Continuation Pointer
15	2	ID	O		0155	00015	Accept Acknowledgment Type
16	2	ID	O		0155	00016	Application Acknowledgment Type
17	2	ID	O			00017	Country Code
18	6	ID	O	Y/3	0211	00692	Character Set
19	60	CE	O			00693	Principal Language Of Message

2.24.1.0 MSH field definitions

2.24.1.1 Field separator (ST) 00001

Definition: This field contains the separator between the segment ID and the first real field, *MSH-2-encoding characters*. As such it serves as the separator and defines the character to be used as a separator for the rest of the message. Recommended value is |, (ASCII 124).

2.24.1.2 Encoding characters (ST) 00002

Definition: This field contains the four characters in the following order: the component separator, repetition separator, escape character, and subcomponent separator. Recommended values are ^~\&, (ASCII 94, 126, 92, and 38, respectively). See Section 2.7, "MESSAGE DELIMITERS."

2.24.1.3 Sending application (HD) 00003

```
Components:  <namespace ID (IS)> ^ <universal ID (ST)> ^ <universal ID type (ID)>
```

Definition: This field uniquely identifies the sending application among all other applications within the network enterprise. The network enterprise consists of all those applications that participate in the exchange of HL7 messages within the enterprise. Entirely site-defined.

2.24.1.4 Sending facility (HD) 00004

```
Components: <namespace ID (IS)> ^ <universal ID (ST)> ^ <universal ID type (ID)>
```

Definition: This field contains the address of one of several occurrences of the same application within the sending system. Absent other considerations, the Medicare Provider ID might be used with an appropriate sub-identifier in the second component. Entirely user-defined.

2.24.1.5 Receiving application (HD) 00005

```
Components: <namespace ID (IS)> ^ <universal ID (ST)> ^ <universal ID type (ID)>
```

Definition: This field uniquely identifies the receiving application among all other applications within the network enterprise. The network enterprise consists of all those applications that participate in the exchange of HL7 messages within the enterprise. Entirely site-defined.

2.24.1.6 Receiving facility (HD) 00006

```
Components: <namespace ID (IS)> ^ <universal ID (ST)> ^ <universal ID type (ID)>
```

Definition: This field identifies the receiving application among multiple identical instances of the application running on behalf of different organizations. See comments: *MSH-4-sending facility*. Entirely site-defined.

2.24.1.7 Date/time of message (TS) 00007

Definition: This field contains the date/time that the sending system created the message. If the time zone is specified, it will be used throughout the message as the default time zone.

2.24.1.8 Security (ST) 00008

Definition: In some applications of HL7, this field is used to implement security features. Its use is not yet further specified.

2.24.1.9 Message type (CM) 00009

```
Components: <message type (ID)> ^ <trigger event (ID)>
```

Definition: This field contains the message type and trigger event for the message. The first component is the message type edited by *HL7 table 0076 - Message type*; second is the trigger event code edited by *HL7 table 0003 - Event type*.

The receiving system uses this field to know the data segments to recognize, and possibly, the application to which to route this message. For certain queries, which may have more than a single response event type, the second component may, in the response message, vary to indicate the response event type. See the discussion of the display query variants in Section 2.17.1.1, "Original mode display query variants." The second component is not required on response or acknowledgment messages.

Table 0076 - Message type

Value	Description	Chapter
ACK	General acknowledgment message	2
ADR	ADT response	3
ADT	ADT message	3
ARD	Ancillary RPT (display) (for backward compatibility only)	7
BAR	Add/change billing account	6
CRM	Clinical study registration	7
CSU	Unsolicited clinical study data	7
DFT	Detail financial transaction	6
DSR	Display response	2
EDR	Enhanced display response	2
ERP	Event replay response	2
EQQ	Embedded query language query	2
MCF	Delayed acknowledgment	2
MDM	Documentation message	9
MFN	Master files notification	8
MFK	Master files application acknowledgment	8
MFD	Master files delayed application acknowledgment	8
MFQ	Master files query	8
MFR	Master files query response	8
ORF	Observ. result/record response	7
ORM	Order message	4
ORR	Order acknowledgment message	4
ORU	Observ result/unsolicited	7
OSQ	Order status query	4
OSR	Order status response	4
PEX	Product experience	7
PGL	Patient goal	12
PIN	Patient insurance information	11
PPG	Patient pathway (goal-oriented) message	12
PPP	Patient pathway (problem-oriented) message	12
PPR	Patient problem	12
PPT	Patient pathway (goal oriented) response	12
PPV	Patient goal response	12
PRR	Patient problem response	12
PTR	Patient pathway (problem-oriented) response	12
QCK	Query general acknowledgment	2
QRY	Query, original Mode	2
RAR	Pharmacy administration information	4
RAS	Pharmacy administration message	4
RCI	Return clinical information	11
RCL	Return clinical list	11
RDE	Pharmacy encoded order message	4
RDR	Pharmacy dispense information	4
RDS	Pharmacy dispense message	4
REF	Patient referral	11
RER	Pharmacy encoded order information	4
RGV	Pharmacy give message	4
RGR	Pharmacy dose information	4
ROR	Pharmacy prescription order response	4

Value	Description	Chapter
RPA	Return patient authorization	11
RPI	Return patient information	11
RPL	Return patient display list	11
RPR	Return patient list	11
RQA	Request patient authorization	11
RQC	Request clinical information	11
RQI	Request patient information	11
RQP	Request patient demographics	11
RRA	Pharmacy administration acknowledgment	4
RRD	Pharmacy dispense acknowledgment	4
RRE	Pharmacy encoded order acknowledgment	4
RRG	Pharmacy give acknowledgment	4
RRI	Return patient referral	11
SIU	Schedule information unsolicited	10
SPQ	Stored procedure request	2
SQM	Schedule query	10
SQR	Schedule query response	10
SRM	Schedule request	10
SRR	Scheduled request response	10
SUR	Summary product experience report	7
TBR	Tabular data response	2
UDM	Unsolicited display message	2
VQQ	Virtual table query	2
VXQ	Query for vaccination record	4
VXX	Vaccination query response with multiple PID matches	4
VXR	Vaccination query record response	4
VXU	Unsolicited vaccination record update	4

Table 0003 - Event type

Value	Description
A01	ADT/ACK - Admit / visit notification
A02	ADT/ACK - Transfer a patient
A03	ADT/ACK - Discharge/end visit
A04	ADT/ACK - Register a patient
A05	ADT/ACK - Pre-admit a patient
A06	ADT/ACK - Change an outpatient to an inpatient
A07	ADT/ACK - Change an inpatient to an outpatient
A08	ADT/ACK - Update patient information
A09	ADT/ACK - Patient departing - tracking
A10	ADT/ACK - Patient arriving - tracking
A11	ADT/ACK - Cancel admit/visit notification
A12	ADT/ACK - Cancel transfer
A13	ADT/ACK - Cancel discharge/end visit
A14	ADT/ACK - Pending admit
A15	ADT/ACK - Pending transfer
A16	ADT/ACK - Pending discharge
A17	ADT/ACK - Swap patients
A18	ADT/ACK - Merge patient information
A19	QRY/ADR - Patient query
A20	ADT/ACK - Bed status update

Value	Description
A21	ADT/ACK - Patient goes on a “leave of absence”
A22	ADT/ACK - Patient returns from a “leave of absence”
A23	ADT/ACK - Delete a patient record
A24	ADT/ACK - Link patient information
A25	ADT/ACK - Cancel pending discharge
A26	ADT/ACK - Cancel pending transfer
A27	ADT/ACK - Cancel pending admit
A28	ADT/ACK - Add person information
A29	ADT/ACK - Delete person information
A30	ADT/ACK - Merge person information
A31	ADT/ACK - Update person information
A32	ADT/ACK - Cancel patient arriving - tracking
A33	ADT/ACK - Cancel patient departing - tracking
A34	ADT/ACK - Merge patient information - patient ID only
A35	ADT/ACK - Merge patient information - account number only
A36	ADT/ACK - Merge patient information - patient ID and account number
A37	ADT/ACK - Unlink patient information
A38	ADT/ACK - Cancel pre-admit
A39	ADT/ACK - Merge person - external ID
A40	ADT/ACK - Merge patient - internal ID
A41	ADT/ACK - Merge account - patient account number
A42	ADT/ACK - Merge visit - visit number
A43	ADT/ACK - Move patient information - internal ID
A44	ADT/ACK - Move account information - patient account number
A45	ADT/ACK - Move visit information - visit number
A46	ADT/ACK - Change external ID
A47	ADT/ACK - Change internal ID
A48	ADT/ACK - Change alternate patient ID
A49	ADT/ACK - Change patient account number
A50	ADT/ACK - Change visit number
A51	ADT/ACK - Change alternate visit ID
C01	CRM - Register a patient on a clinical trial
C02	CRM - Cancel a patient registration on clinical trial (for clerical mistakes only)
C03	CRM - Correct/update registration information
C04	CRM - Patient has gone off a clinical trial
C05	CRM - Patient enters phase of clinical trial
C06	CRM - Cancel patient entering a phase (clerical mistake)
C07	CRM - Correct/update phase information
C08	CRM - Patient has gone off phase of clinical trial
C09	CSU - Automated time intervals for reporting, like monthly
C10	CSU - Patient completes the clinical trial
C11	CSU - Patient completes a phase of the clinical trial
C12	CSU - Update/correction of patient order/result information
CNQ	QRY/EQQ/SPQ/VQQ/RQQ - Cancel query
I01	RQI/RPI - Request for insurance information
I02	RQI/RPL - Request/receipt of patient selection display list
I03	RQI/RPR - Request/receipt of patient selection list
I04	RQD/RPI - Request for patient demographic data
I05	RQC/RCI - Request for patient clinical information
I06	RQC/RCL - Request/receipt of clinical data listing
I07	PIN/ACK - Unsolicited insurance information

Value	Description
I08	RQA/RPA - Request for treatment authorization information
I09	RQA/RPA - Request for modification to an authorization
I10	RQA/RPA - Request for resubmission of an authorization
I11	RQA/RPA - Request for cancellation of an authorization
I12	REF/RRI - Patient referral
I13	REF/RRI - Modify patient referral
I14	REF/RRI - Cancel patient referral
I15	REF/RRI - Request patient referral status
M01	MFN/MFK - Master file not otherwise specified (**for backward compatibility only**)
M02	MFN/MFK - Master file - Staff Practitioner
M03	MFN/MFK - Master file - Test/Observation (**for backward compatibility only**)
varies	MFQ/MFR - Master files query (use event same as asking for e.g., M05 - location)
M04	MFN/MFK - Master files charge description
M05	MFN/MFK - Patient location master file
M06	MFN/MFK - Clinical study with phases and schedules master file
M07	MFN/MFK - Clinical study without phases but with schedules master file
M08	MFN/MFK - Test/observation (Numeric) master file
M09	MFN/MFK - Test/Observation (Categorical) master file
M10	MFN/MFK - Test /observation batteries master file
M11	MFN/MFK - Test/calculated observations master file
O01	ORM - Order message (also RDE, RDS, RGV, RAS)
O02	ORR - Order response (also RRE, RRD, RRG, RRA)
P01	BAR/ACK - Add patient accounts
P02	BAR/ACK - Purge patient accounts
P03	DFT/ACK - Post detail financial transaction
P04	QRY/DSP - Generate bill and A/R statements
P05	BAR/ACK - Update account
P06	BAR/ACK - End account
P07	PEX - Unsolicited initial individual product experience report
P08	PEX - Unsolicited update individual product experience report
P09	SUR - Summary product experience report
PC1	PPR - PC/ Problem Add
PC2	PPR - PC/ Problem Update
PC3	PPR - PC/ Problem Delete
PC4	PRQ - PC/ Problem Query
PC5	PRR - PC/ Problem Response
PC6	PGL - PC/ Goal Add
PC7	PGL - PC/ Goal Update
PC8	PGL - PC/ Goal Delete
PC9	PGQ - PC/ Goal Query
PCA	PGR - PC/ Goal Response
PCB	PPP - PC/ Pathway (Problem-Oriented) Add
PCC	PPP - PC/ Pathway (Problem-Oriented) Update
PCD	PPP - PC/ Pathway (Problem-Oriented) Delete
PCE	PTQ - PC/ Pathway (Problem-Oriented) Query
PCF	PTR - PC/ Pathway (Problem-Oriented) Query Response
PCG	PPG - PC/ Pathway (Goal-Oriented) Add
PCH	PPG - PC/ Pathway (Goal-Oriented) Update
PCJ	PPG - PC/ Pathway (Goal-Oriented) Delete
PCK	PTU - PC/ Pathway (Goal-Oriented) Query
PCL	PTV - PC/ Pathway (Goal-Oriented) Query Response

Value	Description
Q01	QRY/DSR - Query sent for immediate response
Q02	QRY/QCK - Query sent for deferred response
Q03	DSR/ACK - Deferred response to a query
Q05	UDM/ACK - Unsolicited display update message
Q06	OSQ/OSR - Query for order status
R01	ORU/ACK - Unsolicited transmission of an observation message
R02	QRY - Query for results of observation
R03	QRY/DSR Display-oriented results, query/unsol. update **(for backward compatibility only)**
R04	ORF - Response to query; transmission of requested observation
R05	QRY/DSR - query for display results
R06	UDM - unsolicited update/display results
RAR	RAR - Pharmacy administration information query response
RDR	RDR - Pharmacy dispense information query response
RER	RER - Pharmacy encoded order information query response
RGR	RGR - Pharmacy dose information query response
R0R	R0R - Pharmacy prescription order query response
S01	SRM/SRR - Request new appointment booking
S02	SRM/SRR - Request appointment rescheduling
S03	SRM/SRR - Request appointment modification
S04	SRM/SRR - Request appointment cancellation
S05	SRM/SRR - Request appointment discontinuation
S06	SRM/SRR - Request appointment deletion
S07	SRM/SRR - Request addition of service/resource on appointment
S08	SRM/SRR - Request modification of service/resource on appointment
S09	SRM/SRR - Request cancellation of service/resource on appointment
S10	SRM/SRR - Request discontinuation of service/resource on appointment
S11	SRM/SRR - Request deletion of service/resource on appointment
S12	SIU/ACK - Notification of new appointment booking
S13	SIU/ACK - Notification of appointment rescheduling
S14	SIU/ACK - Notification of appointment modification
S15	SIU/ACK - Notification of appointment cancellation
S16	SIU/ACK - Notification of appointment discontinuation
S17	SIU/ACK - Notification of appointment deletion
S18	SIU/ACK - Notification of addition of service/resource on appointment
S19	SIU/ACK - Notification of modification of service/resource on appointment
S20	SIU/ACK - Notification of cancellation of service/resource on appointment
S21	SIU/ACK - Notification of discontinuation of service/resource on appointment
S22	SIU/ACK - Notification of deletion of service/resource on appointment
S23	SIU/ACK - Notification of blocked schedule time slot(s)
S24	SIU/ACK - Notification of opened (“unblocked”) schedule time slot(s)
S25	SQM/SQR - Schedule query message and response
S26	Notification that patient did not show up for schedule appointment
T01	MDM/ACK - Original document notification
T02	MDM/ACK - Original document notification and content
T03	MDM/ACK - Document status change notification
T04	MDM/ACK - Document status change notification and content
T05	MDM/ACK - Document addendum notification
T06	MDM/ACK - Document addendum notification and content
T07	MDM/ACK - Document edit notification
T08	MDM/ACK - Document edit notification and content
T09	MDM/ACK - Document replacement notification

Value	Description
T10	MDM/ACK - Document replacement notification and content
T11	MDM/ACK - Document cancel notification
T12	QRY/DOC - Document query
V01	VXQ - Query for vaccination record
V02	VXX - Response to vaccination query returning multiple PID matches
V03	VXR - Vaccination record response
V04	VXU - Unsolicited vaccination record update
W01	ORU - Waveform result, unsolicited transmission of requested information
W02	QRF - Waveform result, response to query

2.24.1.10 Message control ID (ST) 00010

Definition: This field contains a number or other identifier that uniquely identifies the message. The receiving system echoes this ID back to the sending system in the Message acknowledgment segment (MSA).

2.24.1.11 Processing ID (PT) 00011

```
Components:  <processing ID (ID)> ^ <processing  mode (ID)>
```

Definition: This field is used to decide whether to process the message as defined in HL7 Application (level 7) Processing rules, above. The first component defines whether the message is part of a production, training, or debugging system (refer to *HL7 table 0103 - Processing ID* for valid values). The second component defines whether the message is part of an archival process or an initial load (refer to *HL7 table 0207 - Processing mode* for valid values). This allows different priorities to be given to different processing modes.

Table 0103 - Processing ID

Value	Description
D	Debugging
P	Production
T	Training

Table 0207 - Processing mode

Value	Description
A	Archive
R	Restore from archive
I	Initial load
not present	Not present (the default, meaning *current* processing)

2.24.1.12 Version ID (ID) 00012

Definition: This field is matched by the receiving system to its own version to be sure the message will be interpreted correctly.

Table 0104 - Version ID

Value	Description	
2.0	Release 2.0	September 1988
2.0D	Demo 2.0	October 1988
2.1	Release 2. 1	March 1990
2.2	Release 2.2	December 1994
2.3	Release 2.3	March 1997

2.24.1.13 Sequence number (NM) 00013

Definition: A non-null value in this field implies that the sequence number protocol is in use. This numeric field is incremented by one for each subsequent value.

2.24.1.14 Continuation pointer (ST) 00014

Definition: This field is used to define continuations in application-specific ways.

2.24.1.15 Accept acknowledgment type (ID) 00015

Definition: This field identifies the conditions under which accept acknowledgments are required to be returned in response to this message. Required for enhanced acknowledgment mode. Refer *to HL7 table 0155 - Accept/application acknowledgment conditions* for valid values.

2.24.1.16 Application acknowledgment type (ID) 00016

Definition: This field contains the conditions under which application acknowledgments are required to be returned in response to this message. Required for enhanced acknowledgment mode.

The following table contains the possible values for *MSH-15-accept acknowledgment type* and *MSH-16-application acknowledgment type*:

Table 0155 - Accept/application acknowledgment conditions

Value	Description
AL	Always
NE	Never
ER	Error/reject conditions only
SU	Successful completion only

Note: If *MSH-15-accept acknowledgment type* and *MSH-16-applictation acknowledgment type* are omitted (or are both null), the original acknowledgment mode rules are used.

2.24.1.17 Country code (ID) 00017

Definition: This field contains the country of origin for the message. It will be used primarily to specify default elements, such as currency denominations. ISO 3166 provides a list of country codes that may be used[3].

3 Available from ISO 1 Rue de Varembe, Case Postale 56, CH 1211, Geneve, Switzerland

2.24.1.18 Character set (ID) 00692

Definition: This field contains the character set for the entire message. Refer *to HL7 table 0211 - Alternate character sets* for valid values.

Table 0211 - Alternate character sets

Value	Description
ASCII	The printable 7-bit ASCII character set . (This is the default if this field is omitted)
8859/1	The printable characters from the ISO 8859/1 Character set
8859/2	The printable characters from the ISO 8859/2 Character set
8859/3	The printable characters from the ISO 8859/3 Character set
8859/4	The printable characters from the ISO 8859/4 Character set
8859/5	The printable characters from the ISO 8859/5 Character set
8859/6	The printable characters from the ISO 8859/6 Character set
8859/7	The printable characters from the ISO 8859/7 Character set
8859/8	The printable characters from the ISO 8859/8 Character set
8859/9	The printable characters from the ISO 8859/9 Character set
JAS2020	A subset of ISO2020 used for most Kanjii transmissions
JIS X 0202	ISO 2022 with escape sequences for Kanjii
JIS X 0201-1976	Code for Information Exchange
JIS X 0208-1990	Code for the Japanese Graphic Character set for information interchange
JIS X 0212-1990	Code of the supplementary Japanese Graphic Character set for information interchange

Note: The field separator character must still be chosen from the printable 7-bit ASCII character set.

The repetitions of this field to specify different character sets apply only to fields of the PN and XPN data types.

The field *MSH-18-character set* is an optional, repeating field of data type ID, using IDs outlined in *HL7 table 0211 - Alternate character sets* (or equivalents from ISO 2375).

- if the field is not valued, the default single-byte character set (ASCII (ISO IR-6)) should be assumed. No other character sets are allowed in the message.
- if the field repeats, but the first element is NULL (i.e., present but unvalued), the single-byte ASCII (ISO IR-6) is assumed as the default character set.
- if the sequence is present and the first element is specified, this character set is regarded as the default character set for the message. This must be a single-byte character set (i.e., ISO-IR 6, ISO-IR 13, ISO-IR 14, ISO-IR 100, etc.).

- elements in the remainder of the sequence (i.e., elements 2..n) are alternate character sets that may be used. These may include multi-byte character sets (i.e., JIS X 0208).

- the default character set should always be a single-byte character set. It should always have ISO-IR 6 (ISO 646) or ISO-IR 14 (JIS X 0201-1976) in the G0 area.

2.24.1.19 Principal language of message (CE) 00693

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains the principal language of the message. Codes come from ISO 639.

2.24.2 MSA - message acknowledgment segment

The MSA segment contains information sent while acknowledging another message.

Figure 2-9. MSA attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	2	ID	R		0008	00018	Acknowledgment Code
2	20	ST	R			00010	Message Control ID
3	80	ST	O			00020	Text Message
4	15	NM	O			00021	Expected Sequence Number
5	1	ID	B		0102	00022	Delayed Acknowledgment Type
6	100	CE	O			00023	Error Condition

2.24.2.0 MSA field definitions

2.24.2.1 Acknowledgment code (ID) 00018

Definition: This field contains an acknowledgment code, see message processing rules. Refer to *HL7 table 0008 - Acknowledgment code* for valid values.

Table 0008 - Acknowledgment code

Value	Description
AA	Original mode: Application Accept Enhanced mode: Application acknowledgment: Accept
AE	Original mode: Application Error Enhanced mode: Application acknowledgment: Error
AR	Original mode: Application Reject Enhanced mode: Application acknowledgment: Reject
CA	Enhanced mode: Accept acknowledgment: Commit Accept
CE	Enhanced mode: Accept acknowledgment: Commit Error
CR	Enhanced mode: Accept acknowledgment: Commit Reject

2.24.2.2 Message control ID (ST) 00010

Definition: This field contains the message control ID of the message sent by the sending system. It allows the sending system to associate this response with the message for which it is intended.

2.24.2.3 Text message (ST) 00020

Definition: This optional field further describes an error condition. This text may be printed in error logs or presented to an end user.

2.24.2.4 Expected sequence number (NM) 00021

Definition: This optional numeric field is used in the sequence number protocol.

2.24.2.5 Delayed acknowledgment type (ID) 00022

Definition: ***This field has been retained for backward compatibility.*** This field is used only as described above, in Section 2.12.2, "Application (level 7) processing rules, deferred processing two phase reply (original acknowledgment mode only)." Otherwise this field is not used.

Table 0102 - Delayed acknowledgment type

Value	Description
D	Message received, stored for later processing
F	acknowledgment after processing

2.24.2.6 Error condition (CE) 00023

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field allows the acknowledging system to use a user-defined error code to further specify AR or AE type acknowledgments. This field is a generalized replacement for *MSA-3-text message*.

2.24.3 ERR - error segment

The ERR segment is used to add error comments to acknowledgment messages.

Figure 2-10. ERR attribute

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	80	CM	R	Y		00024	Error Code and Location

2.24.3.0 ERR field definition

2.24.3.1 Error code and location (CM) 00024

```
Components:  <segment ID (ST)> ^ <sequence (NM)> ^ <field position (NM)> ^ <code identifying error (CE)>
```

Definition: This field identifies an erroneous segment in another message. The second component is an index if there is more than one segment of type <segment ID>. For systems that do not use the HL7 Encoding Rules, the data item number may be used for the third component. The fourth component references a locally-defined error table (as a CE data type) is restricted from having any subcomponents as the subcomponent separator is now the CE's component separator.

2.24.4 QRD - original-style query definition segment

The QRD segment is used to define a query.

Figure 2-11. QRD attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	26	TS	R			00025	Query Date/Time
2	1	ID	R		0106	00026	Query Format Code
3	1	ID	R		0091	00027	Query Priority
4	10	ST	R			00028	Query ID
5	1	ID	O		0107	00029	Deferred Response Type
6	26	TS	O			00030	Deferred Response Date/Time
7	10	CQ	R		0126	00031	Quantity Limited Request
8	60	XCN	R	Y		00032	Who Subject Filter
9	60	CE	R	Y	0048	00033	What Subject Filter
10	60	CE	R	Y		00034	What Department Data Code
11	20	ST	O	Y		00035	What Data Code Value Qual.
12	1	ID	O		0108	00036	Query Results Level

2.24.4.0 QRD field definitions

2.24.4.1 Query date/time (TS) 00025

Definition: This field contains the date the query was generated by the application program.

2.24.4.2 Query format code (ID) 00026

Definition: This field refers to *HL7 table 0106 - Query/response format code* for valid values.

Table 0106 - Query/response format code

Value	Description
D	Response is in display format
R	Response is in record-oriented format
T	Response is in tabular format

2.24.4.3 Query priority (ID) 00027

Definition: This field contains the time frame in which the response is expected. Refer to *HL7 table 0091 - Query priority* for valid values. Table values and subsequent fields specify time frames for response.

Table 0091 - Query priority

Value	Description
D	Deferred
I	Immediate

2.24.4.4 Query ID (ST) 00028

Definition: This field contains a unique identifier for the query. Assigned by the querying application. Returned intact by the responding application.

2.24.4.5 Deferred response type (ID) 00029

Definition: This field refers to *HL7 table 0107 - Deferred response type* for valid entries.

Table 0107 - Deferred response type

Value	Description
B	Before the Date/Time specified
L	Later than the Date/Time specified

2.24.4.6 Deferred response date/time (TS) 00030

Definition: This field contains the date/time before or after which to send a deferred response. If not present, the response can be sent when its available. (See *QRD-5-deferred response type* above).

2.24.4.7 Quantity limited request (CQ) 00031

```
Components:  <quantity (NM)> ^ <units (CE)>
```

Definition: This field contains the maximum length of the response that can be accepted by the requesting system. Valid responses are numerical values given in the units specified in the second component. Refer to *HL7 table 0126 - Quantity limited request* for valid entries. Default is LI lines.

Table 0126 - Quantity limited request

Value	Description
CH	Characters
LI	Lines
PG	Pages
RD	Records
ZO	Locally defined

2.24.4.8 Who subject filter (XCN) 00032

```
Components:  <ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^
              <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <source
              table (IS)> ^ <assigning authority (HD)> ^ <name type code (ID)> ^ <code identifying the check
              digit scheme employed(ID)> ^ <identifier type code (IS)> ^ <assigning facility ID (HD)>

Subcomponents of assigning authority:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>

Subcomponents of assigning facility:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>
```

Definition: This field identifies the subject, or who the inquiry is about.

2.24.4.9 What subject filter (CE) 00033

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field describes the kind of information that is required to satisfy the request. Valid values define the type of transaction inquiry and may be extended locally during implementation.

Table 0048 - What subject filter

Value	Description
ADV	Advice/diagnosis
ANU	Nursing unit lookup (returns patients in beds, excluding empty beds)
APN	Patient name lookup
APP	Physician lookup
ARN	Nursing unit lookup (returns patients in beds, including empty beds)
APM	Medical record number query, returns visits for a medical record number
APA	Account number query, return matching visit
CAN	Cancel. Used to cancel a query
DEM	Demographics
FIN	Financial
GOL	Goals
MRI	Most recent inpatient
MRO	Most recent outpatient
NCK	Network clock
NSC	Network status change
NST	Network statistic
ORD	Order
OTH	Other
PRB	Problems
PRO	Procedure
RES	Result
RAR	Pharmacy administration information
RER	Pharmacy encoded order information
RDR	Pharmacy dispense information
RGR	Pharmacy give information
ROR	Pharmacy prescription information
SAL	All schedule related information, including open slots, booked slots, blocked slots
SBK	Booked slots on the identified schedule
SBL	Blocked slots on the identified schedule
SOP	Open slots on the identified schedule
SSA	Time slots available for a single appointment
SSR	Time slots available for a recurring appointment
STA	Status
VXI	Vaccine Information

See the HL7 Implementation Guide for detailed examples of use of various query filter fields.

2.24.4.10 What department data code (CE) 00034

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains the possible contents including test number, procedure number, drug code, item number, order number, etc. The contents of this field are determined by the contents of the previous field. This field could contain multiple occurrences separated by repetition delimiters.

2.24.4.11 What data code value qual (CM) 00035

```
Components:  <first data code value (ST)> ^ <last data code value (ST)>
```

Definition: This field contains the what data code value qualifier. A window or range to further refine the inquiry. This field would contain start/stop separated by component separators.

2.24.4.12 Query results level (ID) 00036

Definition: This field is used to control level of detail in results. Refer to *HL7 table 0108 - Query results level* for valid values. See chapters 4 and 7.

Table 0108 - Query results level

Value	Description
O	Order plus order status
R	Results without bulk text
S	Status only
T	Full results

2.24.5 QRF - original style query filter segment

The QRF segment is used with the QRD segment to further refine the content of an original style query.

Figure 2-12. QRF attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	Item #	Element Name
1	20	ST	R	Y		00037	Where Subject Filter
2	26	TS	O			00038	When Data Start Date/Time
3	26	TS	O			00039	When Data End Date/Time
4	60	ST	O	Y		00040	What User Qualifier
5	60	ST	O	Y		00041	Other QRY Subject Filter
6	12	ID	O	Y	0156	00042	Which Date/Time Qualifier
7	12	ID	O	Y	0157	00043	Which Date/Time Status Qualifier
8	12	ID	O	Y	0158	00044	Date/Time Selection Qualifier
9	60	TQ	O			00694	When Quantity/Timing Qualifier

2.24.5.0 QRF field definitions

2.24.5.1 Where subject filter (ST) 00037

Definition: This field identifies the department, system, or subsystem to which the query pertains. This field may repeat as in LAB~HEMO, etc.

2.24.5.2 When data start date/time (TS) 00038

Definition: This field contains the dates and times equal to or after which this value should be included.

2.24.5.3 When data end date/time (TS) 00039

Definition: This field contains the dates and times equal to or before which this date should be included.

2.24.5.4 What user qualifier (ST) 00040

Definition: This field contains an identifier to further define characteristics of the data of interest.

2.24.5.5 Other QRY subject filter (ST) 00041

Definition: This field contains a filter defined locally for use between two systems. This filter uses codes and field definitions which have specific meaning only to the applications and/or site involved.

2.24.5.6 Which date/time qualifier (ID) 00042

Definition: This field specifies the type of date referred to in *QRF-2-when data start date/time* and *QRF-3-when data end date/time*.

Table 0156 - Which date/time qualifier

Value	Description
ANY	Any date/time within a range
COL	Collection date/time, equivalent to film or sample collection date/time
ORD	Order date/time
RCT	Specimen receipt date/time, receipt of specimen in filling ancillary (Lab)
REP	Report date/time, report date/time at filing ancillary (i.e., Lab)
SCHED	Schedule date/time

2.24.5.7 Which date/time status qualifier (ID) 00043

Definition: This field specifies the status type of objects selected in date range defined by *QRF-2-when data start date/time* and *QRF-3-when data end date/time*).

Table 0157 - Which date/time status qualifier

Value	Description
ANY	Any status
CFN	Current final value, whether final or corrected
COR	Corrected only (no final with corrections)
FIN	Final only (no corrections)
PRE	Preliminary
REP	Report completion date/time

2.24.5.8 Date/time selection qualifier (ID) 00044

Definition: This field allows the specification of certain types of values within the date/time range.

Table 0158 - Date/time selection qualifier

Value	Description
1ST	First value within range
ALL	All values within the range
LST	Last value within the range
REV	All values within the range returned in reverse chronological order (This is the default if not otherwise specified.)

2.24.5.9 When timing/quantity qualifier (TQ) 00694

```
Components:  <quantity (CQ)> ^ <interval (CM)> ^ <duration> ^ <start date/time (TS)> ^ <end date/time (TS)> ^
             <priority (ID)> ^ <condition (ST)> ^ <text (TX)> ^ <conjunction (ID)> ^ <order sequencing>
```

Definition: This field allows an interval definition to be used for specifying multiple responses to a query. With the addition of this filter, new query specifications should no longer use *QRF-2-when data start date/time* and *QRF-3-when data end date/time* in future implementations.

2.24.6 URD - results/update definition segment

The URD segment is used in sending unsolicited updates about orders and results. It's purpose is similar to that of the QRD segment, but from the results/unsolicited update point of view. Some of the fields have parallels in the QRD segment.

Figure 2-13. URD attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	Item #	Element Name
1	26	TS	O			00045	R/U Date/Time
2	1	ID	O		0109	00046	Report Priority
3	60	XCN	R	Y		00047	R/U Who Subject Definition
4	60	CE	O	Y	0048	00048	R/U What Subject Definition
5	60	CE	O	Y		00049	R/U What Department Code
6	20	ST	O	Y		00050	R/U Display/Print Locations
7	1	ID	O		0108	00051	R/U Results Level

2.24.6.0 URD field definitions

2.24.6.1 R/U date/time (TS) 00045

Definition: This field contains the date and time the update was generated by the application program.

2.24.6.2 Report priority (ID) 00046

Definition: This field contains the priority associated with this report or update. Refer to *HL7 table 0109 - Report priority* for valid values.

Table 0109 - Report priority

Value	Description
R	Routine
S	Stat

2.24.6.3 R/U who subject definition (XCN) 00047

```
Components:  <ID number (ST)> ^ <family name (ST)> ^ <given name (ST)> ^ <middle initial or name (ST)> ^
              <suffix (e.g., JR or III) (ST)> ^ <prefix (e.g., DR) (ST)> ^ <degree (e.g., MD) (ST)> ^ <source
              table (IS)> ^ <assigning authority (HD)> ^ <name type code (ID)> ^ <code identifying the check
              digit scheme employed(ID)> ^ <identifier type code (IS)> ^ <assigning facility ID (HD)>

Subcomponents of assigning authority:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>

Subcomponents of assigning facility:  <namespace ID (IS)> & <universal ID (ST)> & <universal ID type (ID)>
```

Definition: This field contains the definition of the subject, or who the report is about.

2.24.6.4 R/U what subject definition (CE) 00048

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field describes the kind of information that is provided in the report. Valid values are the type of transaction inquiry. Refer to *HL7 table 0048 - What subject filter* for valid values.

This table may be extended by local agreement during implementation. See the HL7 Implementation Guide for detailed examples of use of various query filter fields.

2.24.6.5 R/U what department code (CE) 00049

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains either a test number, procedure number, drug code, item number, order number, etc. to identify the department The contents of this field are determined by the contents of the previous field. This field could contain multiple occurrences separated by repetition delimiters.

2.24.6.6 R/U display/print locations (ST) 00050

Definition: This field contains a list of the locations to which the report should be distributed.

2.24.6.7 R/U results level (ID) 00051

Definition: This field is used to control level of detail in results. Refer to *HL7 table 0108 - Query results level* for valid values. Default level is **T** for full results. See chapters 4 and 7.

2.24.7 URS - unsolicited selection segment

The URS segment is identical with the QRF segment, except that if the name of any field contains Query (of QRY), this word has been changed to Results (see *URS-5-R/U other results subject definition*).

Figure 2-14. URS attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	20	ST	R	Y		00052	R/U Where Subject Definition
2	26	TS	O			00053	R/U When Data Start Date/Time
3	26	TS	O			00054	R/U When Data End Date/Time
4	20	ST	O	Y		00055	R/U What User Qualifier
5	20	ST	O	Y		00056	R/U Other Results Subject Definition
6	12	ID	O	Y	0156	00057	R/U Which Date/Time Qualifier
7	12	ID	O	Y	0157	00058	R/U Which Date/Time Status Qualifier
8	12	ID	O	Y	0158	00059	R/U Date/Time Selection Qualifier
9	60	TQ	O			00695	R/U Quantity/Timing Qualifier

2.24.7.0 URS field definitions

2.24.7.1 R/U where subject definition (ST) 00052

Definition: This field identifies the department, system, or subsystem to which the result pertains. This field may repeat as in **LAB~HEMO**, etc.

2.24.7.2 R/U when data start date/time (TS) 00053

Definition: This field contains the date/time the result starts (if applicable).

2.24.7.3 R/U when data end date/time (TS) 00054

Definition: This field contains the date/time the result ends (if applicable).

2.24.7.4 R/U what user qualifier (ST) 00055

Definition: This field contains an identifier to define further the characteristics of the data that are of interest.

2.24.7.5 R/U other results subject definition (ST) 00056

Definition: This field contains a further qualifier, defined locally, for use between two systems. This filter uses codes and field definitions that have specific meaning only to the application and/or site involved.

2.24.7.6 R/U which date/time qualifier (ID) 00057

Definition: This field specifies the type of date referred to in *URS-2-when data start date/time* and *URS-3-when data end date/time*. Refer *to HL7 table 0156 - Which date/time qualifier* for valid values.

2.24.7.7 R/U which date/time status qualifier (ID) 00058

Definition: This field specifies the status type of objects selected in date range defined by *URS-2-when data start date/time* and *URS-3-when data end date/time*. Refer to *HL7 table 0157 - Date/time status qualifier* for valid values.

2.24.7.8 R/U date/time selection qualifier (ID) 00059

Definition: This field allows the specification of certain types of values within the date/time range. Refer to *HL7 table 0158 - Date/time selection qualifier* for valid values.

2.24.7.9 R/U timing/quantity qualifier (TQ) 00695

```
Components:  <quantity (CQ)> ^ <interval (CM)> ^ <duration> ^ <start date/time (TS)> ^ <end date/time (TS)> ^
              <priority (ID)> ^ <condition (ST)> ^ <text (TX)> ^ <conjunction (ID)> ^ <order sequencing>
```

Definition: This field allows an interval definition to be used for specifying multiple responses to a query. With the addition of this filter, new query specifications should no longer use *URS-2-R/U when data start date/time* and *URS-3-R/U when data end date/time* in future implementations

2.24.8 DSC - Continuation pointer segment

The DSC segment is used in the continuation protocol.

Figure 2-15. DSC attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	180	ST	O			00014	Continuation Pointer

2.24.8.0 DSC field definition

2.24.8.1 Continuation pointer (ST) 00014

Definition: This field contains the continuation pointer. See the description of Continuation Fields in Section 2.15.4, "Interactive continuation or cancellation of response messages: original mode (display and record-oriented) and enhanced mode (display, tabular, and event replay)." In an initial query, this field is not present. If the responder returns a value of null or not present, then there is no more data to fulfill any future continuation requests. For use with continuations of unsolicited messages, see Sections 2.14.2, "UDM/ACK - unsolicited display update message (event Q05)," and 2.23.2 "Continuation messages and segments." Note that continuation protocols work with both display- and record-oriented messages.

2.24.9 DSP - display data segment

The DSP segment is used to contain data that has been preformatted by the sender for display. The semantic content of the data is lost; the data is simply treated as lines of text.

Figure 2-16. DSP attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	4	SI	O			00061	Set ID - DSP
2	4	SI	O			00062	Display Level
3	300	TX	R			00063	Data Line
4	2	ST	O			00064	Logical Break Point
5	20	TX	O			00065	Result ID

2.24.9.0 DSP field definitions

2.24.9.1 Set ID - DSP (SI) 00061

Definition: This field is used optionally to number multiple display segments.

2.24.9.2 Display level (SI) 00062

Definition: This field contains numbering to define groups of fields as assigned by the individual sites or applications.

2.24.9.3 Data line (TX) 00063

Definition: This field contains an actual line as it should be displayed. As described for the TX data type, highlighting and other special display characteristics may be included.

2.24.9.4 Logical break point (ST) 00064

Definition: This field is non-null if this line is the last line of a logical break point in the response as defined by the responding system. See Section 2.15.5, "Logical display break points," for the discussion of Logical display break points.

2.24.9.5 Result ID (TX) 00065

Definition: When the user selects a result ID (defined by *DSP-5-logical break point*) from the screen display corresponding to a record in which *DSP-5-result ID* is non-null, the application can initiate a second query (a separate session) to the ancillary with the *QRD-10-what department data code* filled in with this non-null value (e.g., the ancillary accession number or its equivalent). The ancillary response will contain the report referenced by this result ID (e.g., accession number). The ancillary should correlate the result ID with *DSP-4-logical break point* as follows: If more than one line of text is sent per result, *DSP-5-result ID* should be only non-null for a DSP segment that contains a non-null *DSP-4-logical break point*. This field may be broken into components by local agreement. A common example might be to include placer order number, filler order number, and universal service identifier. Whenever such fields are used as components of the result ID, their components will be sent as subcomponents.

2.24.10 ADD - addendum segment

The ADD segment is used to define the continuation of the prior segment in a continuation message. See Section 2.23.2, "Continuation messages and segments," for details.

Figure 2-17. ADD attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1-n	64k	ST	O			00066	Addendum Continuation Pointer

2.24.10.0 ADD field definition

2.24.10.1 Addendum continuation pointer (ST) 00066

Definition: This field is used to define the continuation of the prior segment in a continuation message. See text for details. When the ADD is sent after the segment being continued, it contains no fields. It is only a marker that the previous segment is being continued in a subsequent message. Thus fields 1-N are not present. The sequence designation, 1-N, means the remainder of the fields in the segment being continued. These remainder of the segment being continued fields are present only when the ADD is sent with a continuation message.

2.24.11 FHS - file header segment

The FHS segment is used to head a file (group of batches) as defined in Section 2.23.3, "HL7 batch protocol."

Figure 2-18. FHS attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	1	ST	R			00067	File Field Separator
2	4	ST	R			00068	File Encoding Characters
3	15	ST	O			00069	File Sending Application
4	20	ST	O			00070	File Sending Facility
5	15	ST	O			00071	File Receiving Application
6	20	ST	O			00072	File Receiving Facility
7	26	TS	O			00073	File Creation Date/Time
8	40	ST	O			00074	File Security
9	20	ST	O			00075	File Name/ID
10	80	ST	O			00076	File Header Comment
11	20	ST	O			00077	File Control ID
12	20	ST	O			00078	Reference File Control ID

2.24.11.0 FHS field definitions

2.24.11.1 File field separator (ST) 00067

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.2 File encoding characters (ST) 00068

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.3 File sending application (ST) 00069

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.4 File sending facility (ST) 00070

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.5 File receiving application (ST) 00071

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.6 File receiving facility (ST) 00072

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.7 File creation date/time (TS) 00073

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.8 File security (ST) 00074

Definition: This field has the same definition as the corresponding field in the MSH segment.

2.24.11.9 File name/ID (ST) 00075

Definition: This field can be used by the application processing file. Its use is not further specified.

2.24.11.10 File header comment (ST) 00076

Definition: This field contains the free text field, the use of which is not further specified.

2.24.11.11 File control ID (ST) 00077

Definition: This field is used to identify a particular file uniquely. It can be echoed back in *FHS-12-reference file control ID*.

2.24.11.12 Reference file control ID (ST) 00078

Definition: This field contains the value of *FHS-11-file control ID* when this file was originally transmitted. Not present if this file is being transmitted for the first time.

2.24.12 FTS - file trailer segment

The FTS segment defines the end of a file.

Figure 2-19. FTS attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	10	NM	O			00079	File Batch Count
2	80	ST	O			00080	File Trailer Comment

2.24.12.0 FTS field definitions

2.24.12.1 File batch count (NM) 00079

Definition: This field contains the number of batches contained in this file.

2.24.12.2 File trailer comment (ST) 00080

Definition: The use of this free text field is not further specified.

2.24.13 BHS - batch header segment

The BHS segment defines the start of a batch.

Figure 2-20. BHS attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	1	ST	R			00081	Batch Field Separator
2	3	ST	R			00082	Batch Encoding Characters
3	15	ST	O			00083	Batch Sending Application
4	20	ST	O			00084	Batch Sending Facility
5	15	ST	O			00085	Batch Receiving Application
6	20	ST	O			00086	Batch Receiving Facility
7	26	TS	O			00087	Batch Creation Date/Time
8	40	ST	O			00088	Batch Security
9	20	ST	O			00089	Batch Name/ID/Type
10	80	ST	O			00090	Batch Comment
11	20	ST	O			00091	Batch Control ID
12	20	ST	O			00092	Reference Batch Control ID

2.24.13.0 BHS field definition

2.24.13.1 Batch field separator (ST) 00081

Definition: This field contains the separator between the segment ID and the first real field, *BHS-2-batch encoding characters*. As such it serves as the separator and defines the character to be used as a separator for the rest of the message. Recommended value is |,(ASCII 124).

2.24.13.2 Batch encoding characters (ST) 00082

Definition:. This field contains the four characters in the following order: the component separator, repetition separator, escape characters, and subcomponent separator. Recommended values are ^~\&, (ASCII 94, 126, 92, and 38, respectively. See Section 2.7, "MESSAGE DELIMITERS."

2.24.13.3 Batch sending application (ST) 00083

Definition: This field uniquely identifies the sending application among all other applications within the network enterprise. The network enterprise consists of all those applications that participate in the exchange of HL7 messages within the enterprise. Entirely site-defined.

2.24.13.4 Batch sending facility (ST) 00084

Definition: This field contains the address of one of several occurrences of the same application within the sending system. Absent other considerations, the Medicare Provider ID might be used with an appropriate sub-identifier in the second component. Entirely user-defined.

2.24.13.5 Batch receiving application (ST) 00085

Definition: This field uniquely identifies the receiving applications among all other applications within the network enterprise. The network enterprise consists of all those applications that participate in the exchange of HL7 messages within the enterprise. Entirely site-defined.

2.24.13.6 Batch receiving facility (ST) 00086

Definition: This field identifies the receiving application among multiple identical instances of the application running on behalf of different organizations. See comments *BSH-4-batch sending facility*. Entirely site-defined.

2.24.13.7 Batch creation date/time (TS) 00087

Definition: This field contains the date/time that the sending system created the message. If the time zone is specified, it will be used throughout the message as the default time zone.

2.24.13.8 Batch security (ST) 00088

Definition: In some applications of HL7, this field is used to implement security features. Its use is not yet further specified.

2.24.13.9 Batch name/ID/type (ST) 00089

Definition: This field can be used by the application processing the batch. It can have extra components if needed.

2.24.13.10 Batch comment (ST) 00090

Definition: This field is a comment field that is not further defined in the HL7 protocol.

2.24.13.11 Batch control ID (ST) 00091

Definition: This field is used to uniquely identify a particular batch. It can be echoed back in *BHS-12-reference batch control ID* if an answering batch is needed.

2.24.13.12 Reference batch control ID (ST) 00092

Definition: This field contains the value of *BHS-11-batch control ID* when this batch was originally transmitted. Not present if this batch is being sent for the first time. See definition for *BHS-11-batch control ID*.

2.24.14 BTS - batch trailer segment

The BTS segment defines the end of a batch.

Figure 2-21. BTS attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	10	ST	O			00093	Batch Message Count
2	80	ST	O			00090	Batch Comment
3	100	NM	O	Y		00095	Batch Totals

2.24.14.0 BTS field definitions

2.24.14.1 Batch message count (ST) 00093

Definition: This field contains the count of the individual messages contained within the batch.

2.24.14.2 Batch comment (ST) 00090

Definition: This field is a comment field that is not further defined in the HL7 protocol.

2.24.14.3 Batch totals (NM) 00095

Definition: We encourage new users of this field to use the HL7 Version 2.3 data type of NM and to define it as "repeating." This field contains the batch total. If more than a single batch total exists, this field may be repeated.

This field may be defined as a CM data type for backward compatibility with HL7 Versions 2.2 and 2.1 with each total being carried as a separate component. Each component in this case is an NM data type.

2.24.15 NTE - notes and comments segment

The NTE segment is defined here for inclusion in messages defined in other chapters. It is a common format for sending notes and comments.

Figure 2-22. NTE attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	4	SI	O			00096	Set ID - NTE
2	8	ID	O		0105	00097	Source of Comment
3	64k	FT	O	Y		00098	Comment

2.24.15.0 NTE field definitions

2.24.15.1 Set ID - NTE (SI) 00096

Definition: This field may be used where multiple NTE segments are included in a message. Their numbering must be described in the application message definition.

2.24.15.2 Source of comment (ID) 00097

Definition: This field is used when source of comment must be identified. This table may be extended locally during implementation.

Table 0105 - Source of comment

Value	Description
L	Ancillary (filler) department is source of comment
P	Orderer (placer) is source of comment
O	Other system is source of comment

2.24.15.3 Comment (FT) 00098

Definition: This field contains the comment contained in the segment.

Note: In the current HL7 version, this is an FT rather than a TX data type. Since there is no difference between an FT data type without any embedded formatting commands, and a TX data type, this change is compatible with the previous version.

2.24.16 EQL - embedded query language segment

The EQL segment is used to define queries using select statements based on the query language of choice (e.g., SQL). Refer to the functional chapters for the lists of HL7-defined EQL select statements.

Figure 2-23. EQL attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1	32	ST	O			00696	Query Tag
2	1	ID	R		0106	00697	Query/ Response Format Code
3	60	CE	R			00709	EQL Query Name
4	4096	ST	R			00710	EQL Query Statement

2.24.16.0 EQL field definitions

2.24.16.1 Query tag (ST) 00696

Definition: This field may be valued by the initiating system to identify the query, and may be used to match response messages to the originating query. If it is valued, the responding system is required to echo it back as the first field in the query acknowledgment segment (QAK). This field differs from *MSA-2-message control ID* in that its value remains constant for each message (i.e., all continuation messages) associated with the query, whereas *MSA-2-message control ID* may vary with each continuation message, since it is associated with each individual message, not the query as a whole.

2.24.16.2 Query/response format code (ID) 00697

Definition: This field refers to *HL7 table 0106 - Query/response format code* for valid values.

2.24.16.3 EQL query name (CE) 00709

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains the name of the query. Where the default HL7 coding system is used, these names are assigned by the function-specific chapters of this specification. The values for this field are equivalent to those of *SPR-3-stored procedure name* (see Section 2.24.20, "SPR - stored procedure request definition segment").

2.24.16.4 EQL query statement (ST) 00710

Definition: This field contains the EQL select statement that is the basis of the query.

Fields are designated by the "@" symbol concatenated with the HL7 item number for the field. If the field is divided into components, the designation may be suffixed with ".nn," to identify a particular component (a suffix of ".3" indicates the third component of the field); otherwise, the whole field is assumed. If the field is further divided into subcomponents, the designation is suffixed with ".nn.mm," which identifies the component and subcomponent requested by relative position.

Site-specific fields may be used, provided that they begin with the letter **"Z."** Note that in this case site-defined "z"-item numbers that do not conflict with HL7 items numbers must be negotiated as part of the site specification.

Values for this field are defined in the function-specific chapters of this specification.

Note: If the "@" is being used as one of the delimiter characters defined in *MSH-2-encoding characters*, it must be "escaped ." (See Section 2.9.1, "Formatting codes.")

2.24.17 VTQ - virtual table query request segment

The VTQ segment is used to define queries that are responded to with the Tabular Data Message (TBR). The VTQ query message is an alternate method to the EQQ query message that some systems may find easier to implement, due to its use of HL7 delimiters that separate components of the selection definition, and its limited selection criteria. Queries involving complex selection criteria (nested operators, etc.) may need to be formatted as an EQL segment.

As with the other query methods, the functional chapters define specific queries supported as VTQ messages. Refer to these functional chapters for the lists of HL7-defined virtual tables, selection lists and criteria.

Figure 2-24. VTQ attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1	32	ST	O			00696	Query Tag
2	1	ID	R		0106	00697	Query/Response Format Code
3	60	CE	R			00698	VT Query Name
4	60	CE	R			00699	Virtual Table Name
5	256	QSC	O	Y		00700	Selection Criteria

2.24.17.0 VTQ field definitions

2.24.17.1 Query tag (ST) 00696

Definition: This field may be valued by the initiating system to identify the query, and may be used to match response messages to the originating query. If it is valued, the responding system is required to echo it back as the first field in the query acknowledgment segment (QAK). This field differs from *MSA-2-message control ID* in that its value remains constant for each message (i.e., all continuation messages) associated with the query, whereas *MSA-2-message control ID* may vary with each continuation message, since it is associated with each individual message, not the query as a whole.

2.24.17.2 Query/response format code (ID) 00697

Definition: This field refers to *HL7 table 0106 - Query/response format code* for valid values.

2.24.17.3 VT query name (CE) 00698

```
Components:  <identifier (ST)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ST)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains the name of the virtual table query. These names are assigned by the function-specific chapters of this specification. Site-specific VT query names begin with the letter "Z."

2.24.17.4 Virtual table name (CE) 00699

```
Components:  <identifier (ST)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ST)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains the name of the virtual table being referenced. This table name may refer to an HL7-defined segment, an HL7 virtual table (refer to the functional chapters), or a site-specific "Z table."

2.24.17.5 Selection criteria (QSC) 00700

```
Components:  <name of field (ST)> ^ <relational operator (ID)> ^ <value (ST)> ^ <relational conjunction (ID)>
```

Definition: Each repetition of this field defines a column in the RDT segment: the first repetition defines the first column of the RDT segment; the second repetition defines the second column of the RDT segments, etc.

This field indicates the conditions that qualify the rows to be returned in the query response. (This field conveys the same information as the "WHERE" clause in the corresponding SQL expression of the query, but is formatted differently.) It is comprised of the following components:

- The name of the field that is participating as a qualifier (usually the "key"). Refer to Section 2.24.16.4, "EQL - embedded query language segment," for field naming conventions.

- A relational operator, refer to *HL7 table 0209 - Relational operator* for valid values.

Table 0209 - Relational operator

Relational operator	Value
EQ	Equal
NE	Not Equal
LT	Less than
GT	Greater than
LE	Less than or equal
GE	Greater than or equal
CT	Contains
GN	Generic

- The value to which the field will be compared.

If more than one comparison is to be made to select qualifying rows, a conjunction (defined by *HL7 table 0210 - Relational conjunction*) relating this repetition of the field to the next:

Table 0210 - Relational conjunction

Relational conjunction	Note
AND	Default
OR	

Hence, the segment

```
VTQ|TAG001|T|VT_QUERY_NAME|PID|@00108.1^EQ^EVANS^AND~@00108.2^EQ^CAROLY
N <cr>
```

causes a response to be generated from the virtual table defined by the PID segment. All rows containing the name field subcomponents defined in the selection criteria field (last name = "Evans," first name = "Carolyn") will be selected for the response. The columns returned from each selected row will be defined by the RDF segment (see Section 2.24.18, "RDF - table row definition segment").

Notes:

- As previously stated, the VTQ segment does not, and is not intended to, provide as robust selection function as native EQQ query. It is offered as a simpler alternative.

- When applied to strings, the relational operators LT, GT, LE, and GE imply an alphabetic comparison.
- A "generic" comparison selects a record for inclusion in the response if the beginning of the designated field matches the select string.
- Where a repeating field is specified as an operand, a match on any instance of that field qualifies the row for inclusion in the response message.
- AND takes precedence over OR. More sophisticated precedence rules require that the query be expressed as an SQL message, or a stored procedure for the query may be written and referenced with the SPR segment.

2.24.18 RDF - table row definition segment

The RDF segment defines the content of the row data segments (RDT) in the Tabular Data Response Message (TBR). It is used in two ways:

- As an optional segment in the SPQ message (Stored Procedure Request) or the VQQ (Virtual Table Query) message, this segment can be used to limit the number of columns returned and to specify what column positions the fields occupy (where supported, these features can be used to override the defaults for the particular query). If omitted, all fields defined for the query are returned in their default column order.
- As a required segment on the tabular data response message (TBR), this segment defines the contents of the table row data (RDT) segments that follow.

Figure 2-25. RDF attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1	3	NM	R			00701	Number of Columns per Row
2	40	RCD	R	Y		00702	Column Description

2.24.18.0 RDF field definitions

2.24.18.1 Number of columns per row (NM) 00701

Definition: This field specifies the number of data columns (and therefore the number of fields) contained within each row of returned data.

2.24.18.2 Column description (RCD) 00702

```
Components: <HL7 item number (ST)> ^ <HL7 data type (ST)> ^ <maximum column width (NM)>
```

Definition: Each repetition of this field consists of three components:

- The HL7 item number, which identifies the field occupying the column. (Refer to Section 2.24.16.2, "Query/response format code (ID) 00697," for item numbering conventions).
- The 2 or 3 character HL7 data type, as defined in Section 2.8, "Data types."
- The maximum width of the column, as dictated by the responding system. (This may vary from the HL7-defined maximum field length.)

2.24.19 RDT - table row data segment

The RDT segment contains the row data of the tabular data response message (TBR).

Figure 2-26. RDT attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1-n	Variable	Variable	R			00703	Column Value

2.24.19.0 RDT field definitions

2.24.19.1 Column value (Variable) 00703

Definition: This field is a requested field. Fields occur in the position order defined for the query or table, (unless overridden by an optional RDF segment on a stored procedure request or virtual table query message), separated by field delimiters.

2.24.20 SPR - stored procedure request definition segment

The SPR segment is used to issue queries using stored procedure calls. Refer to the functional chapters for the lists of HL7-defined stored procedure names, input parameters and output tables.

Figure 2-27. SPR attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1	32	ST	O			00696	Query Tag
2	1	ID	R		0106	00697	Query/ Response Format Code
3	60	CE	R			00704	Stored Procedure Name
4	256	QIP	O	Y		00705	Input Parameter List

2.24.20.0 SPR field definitions

2.24.20.1 Query tag (ST) 00696

Definition: This field may be valued by the initiating system to identify the query, and may be used to match response messages to the originating query. If it is valued, the responding system is required to echo it back as the first field in the query acknowledgment segment (QAK). This field differs from *MSA-2-message control ID* in that its value remains constant for each message (i.e., all continuation messages) associated with the query, whereas *MSA-2-message control ID* may vary with each continuation message, since it is associated with each individual message, not the query as a whole.

2.24.20.2 Query/response format code (ID) 00697

Definition: This field refers to *HL7 table 0106 - Query/response format code* for valid values.

Table 0106 - Query/response format code

Value	Description
D	Response is in display format
R	Response is in record-oriented format
T	Response is in tabular format

2.24.20.3 Stored procedure name (CE) 00704

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains the name of the stored procedure that is to be executed. Values for this field are defined in the function-specific chapters of this specification; site-specific stored procedure names begin with the letter **"Z."**

2.24.20.4 Input parameter list (QIP) 00705

```
Components: <field name (ST)> ^ <value1 (ST) & value2 (ST) & value3 (ST) ...>
```

Definition: This field contains the list of parameter names and values to be passed to the stored procedure, in the form "`<field name> ^ <value1& value2 & value3 ...>.`" A single valued parameter contains only a single subcomponent in the second component: thus no subcomponent delimiters are needed (e.g., `<field name> ^ <value>`). A simple list of values (i.e., a one-dimensional array) may be passed instead of a single value by separating each value with the subcomponent delimiter: "`<field name> ^ <value1& value2 &...>`" Refer to Section 2.24.16.4, "EQL query statement (ST) 00710 for field naming conventions.

2.24.21 ERQ - event replay query segment

The ERQ segment is used to issue queries where the desired response is formatted as an event replay response message. This enables the querying application to request detailed event data from an application that supports this feature, such that it may no longer be necessary for it to capture and store all event information at the time of the original trigger event.

Figure 2-28. ERQ attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1	32	ST	O			00696	Query Tag
2	60	CE	R			00706	Event Identifier
3	256	QIP	O	Y		00705	Input Parameter List

2.24.21.0 ERQ field definitions

2.24.21.1 Query tag (ST) 00696

Definition: This field may be valued by the initiating system to identify the query, and may be used to match response messages to the originating query. If it is valued, the responding system is required to echo it back as the first field in the query acknowledgment segment (QAK). This field differs from *MSA-2-message control ID* in that its value remains constant for each message (i.e., all continuation messages) associated with the query, whereas *MSA-2- message control ID* may vary with each continuation message, since it is associated with each individual message, not the query as a whole.

2.24.21.2 Event identifier (CE) 00706

```
Components:  <identifier (ID)> ^ <text (ST)> ^ <name of coding system (ST)> ^ <alternate identifier (ID)> ^
              <alternate text (ST)> ^ <name of alternate coding system (ST)>
```

Definition: This field contains the HL7 event identifier corresponding to the original trigger event. Its contents dictate the format of the response message. Hence, a value of "A04" in this field indicates a request for the data associated with the "register a patient" trigger event. The ERP response message returns the contents of the "register a patient" message defined in Chapter 3. If more than one match is found, the ERP returns repeating groups of the segments defined by the "A04" message.

2.24.21.3 Input parameter list (QIP) 00705

```
Components:  <field name (ST)> ^ <value1 (ST) & value2 (ST) & value3 (ST...>
```

Definition: This field contains the list of parameter names and values to be passed to the responding system, in the form "`<field name> ^ <value1 & value2 & value3 ...>.`" A single valued parameter contains only a single subcomponent in the second component: thus no subcomponent delimiters are needed (e.g., `<field name> ^ <value>`). A simple list of values (i.e., a one-dimensional array) may be passed instead of a single value by separating each value with the subcomponent delimiter: "`<field name> ^ <value1&value2 &...>`" Refer to Section 2.24.16.4, "EQL query statement (ST) 00710," for field naming conventions.

For example, , a value of "@00122^123-45-6789" could be combined with the A04 event identifier to request patient registration data for the patient with the social security number 123-45-6789.

2.24.22 QAK- query acknowledgment segment

The QAK segment contains information sent with responses to a query.

Figure 2-29. QAK attributes

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM#	ELEMENT NAME
1	32	ST	C			00696	Query Tag
2	2	ID	O		0208	00708	Query Response Status

2.24.22.0 QAK field definitions

2.24.22.1 Query tag (ST) 00696

Definition: This field may be valued by the initiating system to identify the query, and may be used to match response messages to the originating query. If it is valued, the responding system is required to echo it back as the first field in the query acknowledgment segment (QAK). This field differs from *MSA-2-message control ID* in that its value remains constant for each message (i.e., all continuation messages) associated with the query, whereas *MSA-2-message control ID* may vary with each continuation message, since it is associated with each individual message, not the query as a whole.

2.24.22.2 Query response status (ID) 00708

Definition: This field allows the responding system to return a precise response status. It is especially useful in the case where no data is found that matches the query parameters, but where there is also no error. It is defined with *HL7 table 0208 - Query response status.*

Table 0208 - Query response status

Value	Description
OK	Data found, no errors (this is the default)
NF	No data found, no errors
AE	Application error
AR	Application reject

2.24.23 Miscellaneous HL7 tables used across all chapters

2.24.23.1 Yes/no indicator table

Table 0136 - Yes/no indicator

Value	Description
Y	Yes
N	No

2.25 SAMPLE CONTROL AND QUERY MESSAGES

2.25.1 General acknowledgment

LAB acknowledges the message that ADT sent identified as ZZ9380. (LAB and ADT, the sending and receiving system IDs, are site-defined.) Both systems are associated with the same FACILITY, 767543. There is no trigger event for an acknowledgment, so the second component of *MSH-9-message type* is not present. The component separator may be present there, but need not be. The **AA** code in the MSA segment indicates that the message was accepted by the application.

```
MSH|^~\&|LAB|767543|ADT|767543|19900314130405||ACK^|XX3657|P|2.1<cr>
MSA|AA|ZZ9380<cr>
```

2.25.2 Error return

The **AR** code in MSA indicates that the application rejected the message for functional reasons. The optional ERR segment includes here that the 16th field of the PID segment with the SET ID value of 1 had an error which was defined by the locally-established code X3L. The optional text message UNKNOWN COUNTY CODE in the link is designed to help programmers and support personnel while reviewing message logs.

```
MSH|^~\&|LAB|767543|ADT|767543|199003141304-0500||ACK^|XX3657|P|2.1<cr>
MSA|AR|ZZ9380|UNKNOWN COUNTY CODE<cr>
ERR|PID^1^16^X3L<cr>
```

2.25.3 Sequence number: initial message

The sender initiates the link with a message that has no functional content. The sequence number is 0. The message type and event code are not used.

```
MSH|^~\&|ADT|767543|LAB|767543|199003141304-0500||^|XX3657|P|2.1|0<cr>
```

The responder uses a general acknowledgment. The expected sequence number is 1.

```
MSH|^~\&|LAB|767543|ADT|767543|199003141304-0500||ACK^|ZZ9380|P|2.1<cr>
MSA|AA|XX3657||1<cr>
```

2.25.4 Query examples

2.25.4.1 Original mode query with display-oriented response

Query for all lab results on patient #12233. The query is made at 11:00 a.m., 9/11/87. The Query anticipates an immediate display-oriented response.

```
MSH|^~\&|ICU||LAB01||||QRY^Q01|MSG00001|P|2.3<cr>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<cr>
```

The response to the above query might look like the following:

```
MSH|^~\&|LAB01||ICU||||DSR|ZXT23461|P|2.3<cr>
MSA|AA|MSG00001P<cr>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<cr>
DSP|||RESULTS FOR PATIENT#12233   SMITH, JOHN H. 09/11/87<cr>
DSP|||SPECIMEN#H85 COLLECTED 09/11/87 /07/0/0<cr>
DSP<cr>
DSP|||ELECTROLYTES<cr>
DSP|||    SODIUM        140 [135-148] MEQ/L  STAT<cr>
DSP|||    POTASSIUM     4.0 [3.5-5.0] MEQ/L  STAT<cr>
DSP|||    CHLORIDE          89  [95-111]      MEQ/L  STAT<cr>
DSP|||    CO2           20  [20-30]       MEQ/L  STAT<cr>
DSP||||LB<cr>
DSP|||CBC<cr>
DSP|||    HEMOGLOBIN        [13.5-18.0]<cr>
DSP|||    HEMATOCRIT    45  [40-54]       %<cr>
DSP|||    RED CELL COUNT    5.0 [4.6-6.2] M/MM3<cr>
DSP|||    MCHC              32  [32-36]       G/DL<cr>
DSP|||    MCH           28  [26-32]       PG<cr>
DSP|||    MCV           85  [81-101]      FL<cr>
DSP|||    WHITE CELL CNT    7.5 [5.0-10.0]K/MM3<cr>
DSP||||LB<cr>
DSP|||SPECIMEN#B24   COLLECTED 9/10/87<cr>
DSC|12333H85;12<cr>
```

A continuation query would echo back the contents of *DSC-1-continuation pointer* as follows:

```
MSH|^~\&|ICU||LAB01||||QRY^Q01|MSG00003|P|2.3<cr>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<cr>
DSC|12333H85;12<cr>
```

The following response shows that there is no further data by leaving *DSC-1-continuation pointer* not present. This could be done by sending the DSC segment with no data, but the example does the same thing by totally omitting the DSC segment.

```
MSH|^~\&|LAB01||ICU|||DSR|ZXT23469|P|2.1<cr>
MSA|AA|MSG00003|<cr>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<cr>
DSP|||RESULTS FOR PATIENT#12233   SMITH, JOHN H. 09/11/87<cr>
DSP|||SPECIMEN#H85 COLLECTED 09/10/87 /07/0/0<cr>
DSP<cr>
DSP|||ELECTROLYTES<cr>
DSP|||    SODIUM    136 [135-148] MEQ/L  STAT<cr>
DSP|||    POTASSIUM 4.2 [3.5-5.0] MEQ/L  STAT<cr>
DSP|||    CHLORIDE      91  [95-111]      MEQ/L  STAT<cr>
DSP|||    CO2       25  [20-30]       MEQ/L  STAT<cr>
DSP||||LB<cr>
```

2.25.4.2 Enhanced mode query examples

Note: For illustration purposes, these examples assume that the following are defined in the ADT chapter:

- The VQQ (using SQL) and EQQ selection criteria
- The virtual table named PID
- The stored procedure named PID_QRY_01

2.25.4.2.1 Embedded query language (using SQL), virtual table and stored procedure queries with tabular response

The following example illustrates a query for the last and first names, address, social security number and date of birth of patients whose last name is "Evans." The fields comprising the query and response are identified by their HL7 item numbers. Where a field is composed of components, the particular component is identified with a "**.n**" suffix (e.g., the patient last name is the first component of the patient name field (HL7 item #00108), and therefore is identified as "@00108.1."

The following examples illustrate this query expressed as an SQL select statement, as a virtual table query and as a stored procedure call

2.25.4.2.2 Embedded query language query

```
MSH|^~\|CLINIC||CENTRAL-REG||||EQQ|MSG00001|P|2.3<cr>
EQL|TAG001|T|SQL_PID_QRY_01|SELECT @00108.1,@00108.2,@00114.1,@00114.2,
   @00114.3,@00114.4,@00114.5,@00122,@00110
   FROM PID WHERE @00108.1='EVANS'<cr>
```

2.25.4.2.3 Virtual table query

```
MSH|^~\|CLINIC||CENTRAL-REG||||VQQ|MSG00001|P|2.3<cr>
VTQ| TAG001|T | VTQ_PID_QRY_01|PID|@00108.1^EQ^EVANS <cr>
RDF|9|@00108.1^ST^20~@00108.2^ST^20~@00114.1^ST^30~@00114.2^ST^30~
   @00114.3^ST^20~@00114.4^ST^2~@00114.5^ST^5~@00122^ST^11~
   @00110^TS^8<cr>
```

2.25.4.2.4 Stored procedure request

```
MSH|^~\|CLINIC||CENTRAL-REG||||SPQ|MSG00001|P|2.3<cr>
SPR|TAG0001|T|SPR_PID_QRY_01|@00108.1^EVANS<cr>
RDF|9|@00108.1^ST^20~@00108.2^ST^20~@00114.1^ST^30~@00114.2^ST^30~
   @00114.3^ST^20~@00114.4^ST^2~@00114.5^ST^5~@00122^ST^11~
   @00110^TS^8<cr>
```

2.25.4.2.5 The response to the above queries might look like the following:

```
MSH|^~\|CENTRAL-REG||CLINIC||||TBR|MSG99001|P|2.3<cr>
MSA|AA|MSG00001<cr>
QAK|TAG0001|OK<cr>
RDF|9|@00108.1^ST^20~@00108.2^ST^20~@00114.1^ST^30~@00114.2^ST^30~
   @00114.3^ST^20~@00114.4^ST^2~@00114.5^ST^5~@00122^ST^11~
   @00110^TS^8<cr>
RDT|Evans|Aaron|105 Maple St.||Lancaster|PA|19786|156-96-
2542|19520809<cr>
RDT|Evans|Bart|166 Norwood Ln.||Hershey|PA|19987|765-58-
4615|19701217<cr>
RDT|Evans|Beth|15 Elmwood Ct.|Apt. 15|Gap|PA|19724|58-96-
7619|19401119<cr>
RDT|Evans|Carolyn|903 Diane Circle||Phoenixville|PA|19460|156-96-
2542|19620324<cr>
DSC|00005<cr>
```

A continuation query would echo back the contents of *DSC-1- continuation pointer*.

2.25.4.2.6 Embedded query language continuation

```
MSH|^~\|CLINIC||CENTRAL-REG||||EQQ|MSG00002|P|2.3<cr>
EQL|TAG001|T|SQL_PID_QRY_01|SELECT @00108.1,@00108.2,@00114.1,@00114.2,
   @00114.3,@00114.4,@00114.5,@00122,@00110
   FROM PID WHERE @00108.1='EVANS'<cr>
DSC|00005<cr>
```

2.25.4.2.7 Virtual table query continuation

```
MSH|^~\|CLINIC||CENTRAL-REG||||VQQ|MSG00002|P|2.3<cr>
VTQ| TAG001|T | VTQ_PID_QRY_01|PID|@00108.1^EQ^EVANS<cr>
RDF|9|@00108.1^ST^20~@00108.2^ST^20~@00114.1^ST^30~@00114.2^ST^30~
   @00114.3^ST^20~@00114.4^ST^2~@00114.5^ST^5~@00122^ST^11~
   @00110^TS^8<cr>
DSC|00005<cr>
```

2.25.4.2.8 Stored procedure request query continuation

```
MSH|^~\|CLINIC||CENTRAL-REG||||SPQ|MSG00002|P|2.3<cr>
SPR|TAG0001|T|SPR_PID_QRY_01|@00108.1^EVANS<cr>
RDF|9|@00108.1^ST^20~@00108.2^ST^20~@00114.1^ST^30~@00114.2^ST^30~
   @00114.3^ST^20~@00114.4^ST^2~@00114.5^ST^5~@00122^ST^11~
   @00110^TS^8<cr>
DSC|00005<cr>
```

2.25.4.2.9 This response shows that there is no further data by leaving the continuation pointer not present. This could be done by sending the DSC segment ID with no data, but the example does the same thing by totally omitting the DSC segment

```
MSH|^~\|CENTRAL-REG||CLINIC||||TBR|MSG00003|P|2.3<cr>
MSA|AA|MSG00002<cr>
QAK|TAG0001|OK<cr>
RDF|9|@00108.1^ST^20~@00108.2^ST^20~@00114.1^ST^30~@00114.2^ST^30~
   @00114.3^ST^20~@00114.4^ST^2~@00114.5^ST^5~@00122^ST^11~
   @00110^TS^8<cr>
RDT|Evans|William|609 N. 3rd St.||Manheim|PA|19898|169-03-
9872|19290726<cr>
RDT|Evans|Zachary|111 North Ln.||Lancaster|PA|19987|539-43-
8725|19340926<cr>
```

2.25.4.2.10 Suppose that from the table of "Evans," Carolyn Evans is selected and the querying application now needs detailed ADT information about her. It can issue another query for this information using the event replay query (EQQ).

```
MSH|^~\|CLINIC||CENTRAL-REG||||RQQ|MSG00004|P|2.3<cr>

ERQ|TAG0002|A04|@00122^156-96-2542<cr>
```

2.25.4.2.11 The response is returned as an Event Replay Response, which is the HL7 ADT patient registration message corresponding to event code A04, prefixed by the MSH, MSA and ERQ segments:

```
MSH|^~\|CLINIC||CENTRAL-REG||||ERP|MSG00005|P|2.3<cr>

MSA|AA|MSG00004<cr>

ERQ|TAG0002|A04|@00122^156-96-2542<cr>

QAK|TAG0002|OK<cr>

EVN|A04|199405151259||<cr>

PID|||2-68708-5|253763|EVANS^CAROLYN||19620324|F|||903
   Diane Circle^^PHOENIXVILLE^PA^19460|(610)555-1212|
   (610)555-1212||S|C||156-96-2542||<cr>

NK1||EVANS^RICHARD|SPOUSE|903Diane Circle^^PHOENIXVILLE^
   PA^19460|(610)555-1212|<cr>

PV1||E|EMERG||||0148^ADDISON^JAMES<cr>

..etc
```

Error responses to the above queries might look like the following:

2.25.4.2.12 Embedded query language (EQL) , virtual table, and stored procedure error response

```
MSH|^~\|CENTRAL-REG||CLINIC||||TBR|MSG99001|P|2.3<cr>

MSA|AE|MSG00001||||^REQUESTED TABLE “PID” IS UNKNOWN<cr>

QAK|TAG0001|AE<cr>
```

2.25.4.2.13 Event replay error response

```
MSH|^~\|CENTRAL-REG||CLINIC||||ERP|MSG00005|P|2.3<cr>

MSA|AE|MSG00004||||^REQUESTED EVENT TYPE “A04” NOT SUPPORTED ON THIS
SYSTEM<cr>

QAK|TAG0002|AE<cr>
```

2.25.5 Master file update examples: with original and enhanced acknowledgment protocol

This example shows the lab system using the Master Files specification to send two update test dictionary entries to an ICU system. The OM1 (observation dictionary) segment, currently under development by HL7 and ASTM, carries the dictionary information. Several varieties of acknowledgment are shown. The choice of acknowledgment mode is site-specific.

2.25.5.1 Original mode example:

```
MSH|^~\&|LABxxx|ClinLAB|ICU||19910918060544||MFN^M03|MSGID002|P|2.2
MFI|LABxxx^Lab Test Dictionary^L|UPD|||AL
MFE|MUP|199109051000|199110010000|12345^WBC^L
OM1|...
MFE|MUP|199109051015|199110010000|6789^RBC^L
OM1|...
```

Original mode acknowledgment of the HL7 message according to MFI Response Level Code of AL.

```
MSH|^~\&|ICU||LABxxx|ClinLAB|19910918060545||MFK|MSGID99002|P|2.2
MSA|AA|MSGID002
MFI|LABxxx^Lab Test Dictionary^L|UPD|||MFAA
MFA|MUP|199110010000|199110010040|S|12345^WBC^L
MFA|MUP|199110010000|199110010041|S|6789^RBC^L
```

2.25.5.2 Enhanced mode example

2.25.5.2.1 Initial message with accept acknowledgment

```
MSH|^~\&|LABxxx|ClinLAB|ICU||19910918060544||MFN^M03|MSGID002|P|2.2|||A
L|AL
MFI|LABxxx^Lab Test Dictionary^L|UPD|||AL
MFE|MUP|199109051000|199110010000|12345^WBC^L
OM1|...
MFE|MUP|199109051015|199110010000|6789^RBC^L
OM1|...
```

```
MSH|^~\&|ICU||LABxxx|ClinLAB|19910918060545||MSA|MSGID99002|P|2.2
MSA|CA|MSGID002
```

2.25.5.2.2 Application acknowledgment message

```
MSH|^~\&|ICU||LABxxx|ClinLAB|19911001080504||MFK|MSGID5002|P|2.2|||AL|
MSA|AA|MSGID002
MFI|LABxxx^Lab Test Dictionary^L|UPD|||MFAA
MFA|MUP|199109051000|199110010040|S|12345^WBC^L
MFA|MUP|199109051015|199110010041|S|6789^RBC^L
```

```
MSH|^~\&|LABxxx|ClinLAB|ICU||19911001080507||ACK|MSGID444|P|2.2

MSA|CA|MSGID5002
```

2.25.5.3 Delayed application acknowledgment

> **Note:** If the MFN message in Section 2.25.5.1, "Original mode example:," had not required an application acknowledgment at the message level (i.e., the application acknowledgment code of the MSH segment = NE), the (Master Files Chapter defined) MFD message could be used to provide a delayed application level acknowledgment not tied to the original MFN message.

The following example includes an acknowledgment for an MFE segment not in the original message. This additional MFE was sent via another MFN message.

2.25.5.3.1 Initial message with accept acknowledgment

```
MSH|^~\&|LABxxx|ClinLAB|ICU||19910918060544||MFN^M03|MSGID002|P|2.2|||A
L|NE

MFI|LABxxx^Lab Test Dictionary^L|UPD|||AL

MFE|MUP|199109051000|199110010000|12345^WBC^L

OM1|...

MFE|MUP|199109051015|199110010000|6789^RBC^L

OM1|...


MSH|^~\&|ICU||LABxxx|ClinLAB|19910918060545||MSA|MSGID99002|P|2.2

MSA|CA|MSGID002
```

2.25.5.3.2 Delayed application acknowledgment

```
MSH|^~\&|ICU||LABxxx|ClinLAB|19911001080504||MFD|MSGID65002|P|2.2|||AL|

MFI|LABxxx^Lab Test Dictionary^L|UPD|||MFAA

MFA|MUP|199109051000|199110010040|S|12345^WBC^L

MFA|MUP|199109051015|199110010041|S|6789^RBC^L

MFA|MUP|199109051025|199110010041|S|4339^HGB^L


MSH|^~\&|LABxxx|ClinLAB|ICU||19911001080507||ACK|MSGID444|P|2.2

MSA|CA|MSGID65002
```

2.26 OUTSTANDING ISSUES

The following items are being discussed in the Control/Query technical committee for addition to future versions of HL7.

1. Rationalization and clarification of event structures.

2. Creation of a network server for HL7 tables so that updates to them can be made public immediately, rather than waiting until the publication of the next version of the Standard.

3. Extensions to the encoding rules for Version 3.

4. Consideration of security. There are in general two types: application level security, which is partially addressed by the *security* field in the MSH segment. The second type, network security, needs to be addressed in the HL7 Implementation Guide. There are several commercially available encryption-based approaches to network level security.