```
Z39-50-APDU-1995 -- OID for this definition, assigned in OID.3.1, is {Z39-50 2 1}
DEFINITIONS ::=
BEGIN   -- Z39.50 Maintenance Agency Official Text for ANSI/NISO Z39.50-1995 - July 1995
--
EXPORTS OtherInformation, Term, AttributeSetId, AttributeList, AttributeElement, ElementSetName, SortElement,
DatabaseName, CompSpec, Specification, Permissions, InternationalString, IntUnit, Unit, StringOrNumeric, Query,
Records, ResultSetId, DefaultDiagFormat, DiagRec;
--
PDU ::= CHOICE{
 initRequest                   [20] IMPLICIT InitializeRequest,
 initResponse                  [21] IMPLICIT InitializeResponse,
 searchRequest                 [22] IMPLICIT SearchRequest,
 searchResponse                [23] IMPLICIT SearchResponse,
 presentRequest                [24] IMPLICIT PresentRequest,
 presentResponse               [25] IMPLICIT PresentResponse,
 deleteResultSetRequest        [26] IMPLICIT DeleteResultSetRequest,
 deleteResultSetResponse       [27] IMPLICIT DeleteResultSetResponse,
 accessControlRequest          [28] IMPLICIT AccessControlRequest,
 accessControlResponse         [29] IMPLICIT AccessControlResponse,
 resourceControlRequest        [30] IMPLICIT ResourceControlRequest,
 resourceControlResponse       [31] IMPLICIT ResourceControlResponse,
 triggerResourceControlRequest [32] IMPLICIT TriggerResourceControlRequest,
 resourceReportRequest         [33] IMPLICIT ResourceReportRequest,
 resourceReportResponse        [34] IMPLICIT ResourceReportResponse,
 scanRequest                   [35] IMPLICIT ScanRequest,
 scanResponse                  [36] IMPLICIT ScanResponse,
                                     -- [37] through [42] reserved
 sortRequest                   [43] IMPLICIT SortRequest,
 sortResponse                  [44] IMPLICIT SortResponse,
 segmentRequest                [45] IMPLICIT Segment,
 extendedServicesRequest       [46] IMPLICIT ExtendedServicesRequest,
 extendedServicesResponse      [47] IMPLICIT ExtendedServicesResponse,
 close                         [48] IMPLICIT Close}


-- Initialize APDUs
--
```

```
  InitializeRequest ::= SEQUENCE{
        referenceId                          ReferenceId OPTIONAL,
        protocolVersion                      ProtocolVersion,
        options                              Options,
        preferredMessageSize        [5]      IMPLICIT INTEGER,
        exceptionalRecordSize       [6]      IMPLICIT INTEGER,
        idAuthentication            [7]      ANY OPTIONAL, -- see note below
        implementationId            [110]    IMPLICIT InternationalString OPTIONAL,
        implementationName          [111]    IMPLICIT InternationalString OPTIONAL,
        implementationVersion       [112]    IMPLICIT InternationalString OPTIONAL,
        userInformationField        [11]     EXTERNAL OPTIONAL,
        otherInfo                            OtherInformation OPTIONAL}
--Note:
-- For idAuthentication, the type ANY is retained for compatibility with earlier versions.
-- For interoperability, the following is recommended:
--   IdAuthentication [7] CHOICE{
--       open    VisibleString,
--       idPass  SEQUENCE {
--                   groupId     [0]     IMPLICIT InternationalString OPTIONAL,
--                   userId      [1]     IMPLICIT InternationalString OPTIONAL,
--                   password    [2]     IMPLICIT InternationalString OPTIONAL },
--       anonymous   NULL,
--       other       EXTERNAL
-- May use access control formats for 'other'.  See Appendix 7 ACC.
--
  InitializeResponse ::= SEQUENCE{
        referenceId                          ReferenceId OPTIONAL,
        protocolVersion                      ProtocolVersion,
        options                                      Options,
        preferredMessageSize        [5]      IMPLICIT INTEGER,
        exceptionalRecordSize       [6]      IMPLICIT INTEGER,
        result                      [12]     IMPLICIT BOOLEAN,    -- reject = FALSE; Accept = TRUE
        implementationId            [110]    IMPLICIT InternationalString OPTIONAL,
        implementationName          [111]    IMPLICIT InternationalString OPTIONAL,
        implementationVersion       [112]    IMPLICIT InternationalString OPTIONAL,
        userInformationField        [11]     EXTERNAL OPTIONAL,
        otherInfo                            OtherInformation OPTIONAL}
-- Begin auxiliary definitions for Init PDUs
  ProtocolVersion  ::=       [3]   IMPLICIT BIT STRING{
              version-1                      (0),     -- This bit should always be set, but does not
                                                      -- correspond to any Z39.50 version.
              version-2                      (1),     -- "Version 2 supported."
                                                      -- This bit should always be set.
              version-3                      (2)      -- "Version 3 supported."
-- Values higher than 'version-3' should be ignored. Both the Initialize request and Initialize Response APDUs
-- include a value string corresponding to the supported versions. The highest common version is selected
-- for use. If there are no versions in common, "Result" in the Init Response should indicate "reject."
-- Note: Versions 1 and 2 are identical. Systems supporting version 2 should indicate support for version
-- 1 as well, for interoperability with systems that indicate support for version 1 only (e.g. ISO 10163-1991
-- implementations).
        }
```

```
  Options  ::= [4] IMPLICIT BIT STRING{
                        search                    (0),
                        present                   (1),
                        delSet                    (2),
                        resourceReport            (3),
                        triggerResourceCtrl       (4),
                        resourceCtrl              (5),
                        accessCtrl                (6),
                        scan                      (7),
                        sort                      (8),
                        --                        (9) (reserved)
                        extendedServices          (10),
                        level-1Segmentation       (11),
                        level-2Segmentation       (12),
                        concurrentOperations      (13),
                        namedResultSets           (14)}
-- end auxiliary definitions for Init PDUs


--Search APDUs
  SearchRequest ::= SEQUENCE{
        referenceId                              ReferenceId OPTIONAL,
        smallSetUpperBound            [13]       IMPLICIT INTEGER,
        largeSetLowerBound            [14]       IMPLICIT INTEGER,
        mediumSetPresentNumber        [15]       IMPLICIT INTEGER,
        replaceIndicator              [16]       IMPLICIT BOOLEAN,
        resultSetName                 [17]       IMPLICIT InternationalString,
        databaseNames                 [18]       IMPLICIT SEQUENCE OF DatabaseName,
        smallSetElementSetNames       [100]      ElementSetNames OPTIONAL,
        mediumSetElementSetNames      [101]      ElementSetNames OPTIONAL,
        preferredRecordSyntax         [104]      IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        query                         [21]       Query,
              -- Following two parameters may be used only if version 3 is in force.
        additionalSearchInfo          [203]      IMPLICIT OtherInformation OPTIONAL,
        otherInfo                                OtherInformation OPTIONAL}


-- Query Definitions
        Query  ::=      CHOICE{
                        type-0        [0]      ANY,
                        type-1        [1]      IMPLICIT RPNQuery,
                        type-2        [2]      OCTET STRING,
                        type-100      [100]    OCTET STRING,
                        type-101      [101]    IMPLICIT RPNQuery,
                        type-102      [102]    OCTET STRING}
--
-- Definitions for RPN query
        RPNQuery ::= SEQUENCE{
                        attributeSet           AttributeSetId,
                        rpn                    RPNStructure}
--
```

```
 RPNStructure ::= CHOICE{
          op            [0] Operand,
          rpnRpnOp      [1] IMPLICIT SEQUENCE{
                              rpn1          RPNStructure,
                              rpn2          RPNStructure,
                              op            Operator }}
     Operand ::= CHOICE{
          attrTerm      AttributesPlusTerm,
          resultSet     ResultSetId,
                              -- If version 2 is in force:
                              --   - If query type is 1, one of the above two must be chosen;
                              --   - resultAttr (below) may be used only if query type is 101.
          resultAttr    ResultSetPlusAttributes}

     AttributesPlusTerm ::= [102] IMPLICIT SEQUENCE{
                        attributes      AttributeList,
                        term            Term}
     ResultSetPlusAttributes ::= [214] IMPLICIT SEQUENCE{
                        resultSet       ResultSetId,
                        attributes      AttributeList}
     AttributeList ::=              [44]  IMPLICIT SEQUENCE OF AttributeElement
--
     Term ::= CHOICE{
          general                   [45]    IMPLICIT OCTET STRING,
                                    -- values below may be used only if version 3 is in force
          numeric                   [215]  IMPLICIT INTEGER,
          characterString           [216]  IMPLICIT InternationalString,
          oid                       [217]  IMPLICIT OBJECT IDENTIFIER,
          dateTime                  [218]  IMPLICIT GeneralizedTime,
          external                  [219]  IMPLICIT EXTERNAL,
          integerAndUnit            [220] IMPLICIT IntUnit,
          null                      [221] IMPLICIT NULL}

     Operator ::= [46] CHOICE{
                         and        [0] IMPLICIT NULL,
                         or         [1] IMPLICIT NULL,
                         and-not    [2] IMPLICIT NULL,
                                          -- If version 2 is in force:
                                          --  - For query type 1, one of the above three must be chosen;
                                          --  - prox (below) may be used only if query type is 101.
                        prox        [3] IMPLICIT ProximityOperator}
     AttributeElement  ::=  SEQUENCE{
          attributeSet          [1]           IMPLICIT AttributeSetId OPTIONAL,
                                                  -- Must be omitted if version 2 is in force.
                                                  -- If included, overrides value of attributeSet
                                                  -- in RPNQuery above, but only for this attribute.
          attributeType [120]  IMPLICIT INTEGER,
          attributeValue                CHOICE{
                                          numeric      [121]   IMPLICIT INTEGER,
                                                         -- If version 2 is in force,
                                                         -- Must select 'numeric' for attributeValue.
```

```
            complex     [224] IMPLICIT SEQUENCE{
                                list                [1] IMPLICIT SEQUENCE OF StringOrNumeric,
                                semanticAction      [2] IMPLICIT SEQUENCE OF INTEGER OPTIONAL}}}


      ProximityOperator ::= SEQUENCE{
            exclusion               [1] IMPLICIT BOOLEAN OPTIONAL,
            distance                [2] IMPLICIT INTEGER,
            ordered                 [3] IMPLICIT BOOLEAN,
            relationType            [4] IMPLICIT INTEGER{
                                          lessThan                (1),
                                          lessThanOrEqual         (2),
                                          equal                   (3),
                                          greaterThanOrEqual      (4),
                                          greaterThan             (5),
                                          notEqual                (6)},
            proximityUnitCode       [5] CHOICE{
                                          known       [1] IMPLICIT KnownProximityUnit,
                                          private     [2] IMPLICIT INTEGER}}
--
   KnownProximityUnit ::= INTEGER{
                                character     (1),
                                word          (2),
                                sentence      (3),
                                paragraph     (4),
                                section       (5),
                                chapter       (6),
                                document      (7),
                                element       (8),
                                subelement    (9),
                                elementType   (10),
                                byte          (11) -- Version 3 only
                                }
-- End definitions for RPN Query


SearchResponse ::= SEQUENCE{
      referenceId                        ReferenceId OPTIONAL,
      resultCount               [23]     IMPLICIT INTEGER,
      numberOfRecordsReturned   [24]     IMPLICIT INTEGER,
      nextResultSetPosition     [25]     IMPLICIT INTEGER,
      searchStatus              [22]     IMPLICIT BOOLEAN,
      resultSetStatus           [26]     IMPLICIT INTEGER{
                                               subset      (1),
                                               interim     (2),
                                               none        (3)} OPTIONAL,
      presentStatus                      PresentStatus  OPTIONAL,
      records                            Records OPTIONAL,
            -- Following two parameters may be used only if version 3 is in force.
      additionalSearchInfo      [203]    IMPLICIT OtherInformation OPTIONAL,
      otherInfo                          OtherInformation OPTIONAL}
```

```
--Retrieval APDUs
  PresentRequest ::= SEQUENCE{
        referenceId                                   ReferenceId OPTIONAL,
        resultSetId                                   ResultSetId,
        resultSetStartPoint             [30]    IMPLICIT INTEGER,
        numberOfRecordsRequested        [29]    IMPLICIT INTEGER,
        additionalRanges                [212]   IMPLICIT SEQUENCE OF Range OPTIONAL,
                -- additionalRanges may be included only if version 3 is in force.
        recordComposition               CHOICE{
                                                simple          [19]    ElementSetNames,
                                                -- must choose 'simple' if version 2 is in force
                                                complex         [209]   IMPLICIT CompSpec} OPTIONAL,
        preferredRecordSyntax           [104]   IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        maxSegmentCount                 [204]   IMPLICIT INTEGER OPTIONAL, -- level 1 or 2
        maxRecordSize                   [206]   IMPLICIT INTEGER OPTIONAL, -- level 2 only
        maxSegmentSize                  [207]   IMPLICIT INTEGER OPTIONAL, -- level 2 only
        otherInfo                               OtherInformation OPTIONAL}
--
  Segment ::= SEQUENCE{
                -- Segment PDU may only be used when version 3 is in force,
                -- and only when segmentation is in effect.
        referenceId                             ReferenceId OPTIONAL,
        numberOfRecordsReturned         [24]    IMPLICIT INTEGER,
        segmentRecords                  [0]     IMPLICIT SEQUENCE OF NamePlusRecord,
        otherInfo                               OtherInformation OPTIONAL}
--
  PresentResponse ::= SEQUENCE{
        referenceId                             ReferenceId OPTIONAL,
        numberOfRecordsReturned         [24]    IMPLICIT INTEGER,
        nextResultSetPosition           [25]    IMPLICIT INTEGER,
        presentStatus                           PresentStatus,
        records                                 Records OPTIONAL,
        otherInfo                               OtherInformation OPTIONAL}
-- begin auxiliary definitions for Search and Present APDUs

-- begin definition of records
  Records ::= CHOICE{
        responseRecords                 [28]    IMPLICIT SEQUENCE OF NamePlusRecord,
        nonSurrogateDiagnostic          [130]   IMPLICIT DefaultDiagFormat,
        multipleNonSurDiagnostics       [205]   IMPLICIT SEQUENCE OF DiagRec}
--
  NamePlusRecord  ::=  SEQUENCE{
        name            [0] IMPLICIT DatabaseName OPTIONAL,
        record          [1] CHOICE{
                        retrievalRecord         [1] EXTERNAL,
                        surrogateDiagnostic     [2] DiagRec,
                                -- Must select one of the above two, retrievalRecord or
                                -- surrogateDiagnostic, unless 'level 2 segmentation' is in effect.
                        startingFragment        [3] FragmentSyntax,
                        intermediateFragment    [4] FragmentSyntax,
                        finalFragment           [5] FragmentSyntax}}
```

```
    FragmentSyntax ::= CHOICE{
        externallyTagged            EXTERNAL,
        notExternallyTagged         OCTET STRING}

 DiagRec ::= CHOICE{
            defaultFormat               DefaultDiagFormat,
                                    -- Must choose defaultFormat if version 2 is in effect.
            externallyDefined           EXTERNAL}

 DefaultDiagFormat::= SEQUENCE{
    diagnosticSetId     OBJECT IDENTIFIER,
    condition           INTEGER,
    addinfo             CHOICE{
                            v2Addinfo   VisibleString,      -- version 2
                            v3Addinfo   InternationalString -- version 3
                             }}
 -- end definition of records
    Range  ::= SEQUENCE{
        startingPosition            [1] IMPLICIT INTEGER,
        numberOfRecords             [2] IMPLICIT INTEGER}
--
    ElementSetNames ::= CHOICE {
        genericElementSetName       [0] IMPLICIT InternationalString,
        databaseSpecific            [1] IMPLICIT SEQUENCE OF SEQUENCE{
                                        dbName      DatabaseName,
                                        esn         ElementSetName}}

    PresentStatus       ::=     [27]    IMPLICIT INTEGER{
                                        success     (0),
                                        partial-1   (1),
                                        partial-2   (2),
                                        partial-3   (3),
                                        partial-4   (4),
                                        failure     (5)}

-- begin definition of composition specification
 CompSpec ::= SEQUENCE{
    selectAlternativeSyntax     [1] IMPLICIT BOOLEAN,
                                    -- See comment for recordSyntax, below.
    generic                     [2] IMPLICIT Specification OPTIONAL,
    dbSpecific                  [3] IMPLICIT SEQUENCE OF SEQUENCE{
                                        db      [1] DatabaseName,
                                        spec    [2] IMPLICIT Specification} OPTIONAL,
        -- At least one of generic and dbSpecific must occur, and both may occur. If both, then for
        -- any record not in the list of databases within dbSpecific, generic applies.
    recordSyntax                [4] IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL
                                    -- For each record, the target selects the first record syntax
                                    -- in this list that it can support.  If the list is exhausted, the
                                    -- target may select an alternative syntax if
                                    -- selectAlternativeSyntax is 'true'.
                        }
```

```
  Specification ::= SEQUENCE{
        schema              [1] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        elementSpec         [2] CHOICE{
                                    elementSetName     [1] IMPLICIT InternationalString,
                                    externalEspec      [2] IMPLICIT EXTERNAL} OPTIONAL}
-- end definition of composition specification
-- end auxiliary definitions for search and response APDUs

-- Delete APDUs
  DeleteResultSetRequest ::= SEQUENCE{
        referenceId                            ReferenceId OPTIONAL,
        deleteFunction                  [32]   IMPLICIT INTEGER{
                                                   list    (0),
                                                   all     (1)},
        resultSetList                          SEQUENCE OF ResultSetId OPTIONAL,
        otherInfo                              OtherInformation OPTIONAL}
--
  DeleteResultSetResponse ::= SEQUENCE{
        referenceId                            ReferenceId OPTIONAL,
        deleteOperationStatus           [0]    IMPLICIT DeleteSetStatus,
        deleteListStatuses              [1]    IMPLICIT ListStatuses OPTIONAL,
        numberNotDeleted                [34]   IMPLICIT INTEGER OPTIONAL,
        bulkStatuses                    [35]   IMPLICIT ListStatuses OPTIONAL,
        deleteMessage                   [36]   IMPLICIT InternationalString OPTIONAL,
        otherInfo                              OtherInformation OPTIONAL}
        ListStatuses ::= SEQUENCE OF SEQUENCE{
                            id       ResultSetId,
                            status   DeleteSetStatus}

        DeleteSetStatus ::= [33] IMPLICIT INTEGER{
                    success                               (0),
                    resultSetDidNotExist                  (1),
                    previouslyDeletedByTarget             (2),
                    systemProblemAtTarget                 (3),
                    accessNotAllowed                      (4),
                    resourceControlAtOrigin               (5),
                    resourceControlAtTarget               (6),
                    bulkDeleteNotSupported                (7),
                    notAllRsltSetsDeletedOnBulkDlte       (8),
                    notAllRequestedResultSetsDeleted      (9),
                    resultSetInUse                        (10)}
--

--Access- and Resource-control APDUs
--
  AccessControlRequest ::= SEQUENCE{
        referenceId                     ReferenceId OPTIONAL,
        securityChallenge               CHOICE{
                                            simpleForm            [37] IMPLICIT OCTET STRING,
                                            externallyDefined     [0]  EXTERNAL},
        otherInfo                       OtherInformation OPTIONAL}
```

```
AccessControlResponse ::= SEQUENCE{
      referenceId                         ReferenceId OPTIONAL,
      securityChallengeResponse  CHOICE{
                                              simpleForm              [38]    IMPLICIT OCTET STRING,
                                              externallyDefined       [0]     EXTERNAL} OPTIONAL,
                                              -- Optional only in version 3; mandatory in version 2. If
                                              -- omitted (in version 3) then diagnostic must occur.
      diagnostic              [223]       DiagRec OPTIONAL, -- Version 3 only.
      otherInfo                           OtherInformation OPTIONAL}



ResourceControlRequest ::= SEQUENCE{
      referenceId                             ReferenceId OPTIONAL,
      suspendedFlag                   [39]    IMPLICIT BOOLEAN OPTIONAL,
      resourceReport                  [40]    ResourceReport OPTIONAL,
      partialResultsAvailable         [41]    IMPLICIT INTEGER{
                                                      subset          (1),
                                                      interim         (2),
                                                      none            (3)} OPTIONAL,
      responseRequired                [42]    IMPLICIT BOOLEAN,
      triggeredRequestFlag            [43]    IMPLICIT BOOLEAN OPTIONAL,
      otherInfo                               OtherInformation OPTIONAL}


ResourceControlResponse ::= SEQUENCE{
      referenceId                             ReferenceId OPTIONAL,
      continueFlag                    [44]    IMPLICIT BOOLEAN,
      resultSetWanted                 [45]    IMPLICIT BOOLEAN OPTIONAL,
      otherInfo                               OtherInformation OPTIONAL}



TriggerResourceControlRequest ::= SEQUENCE{
      referenceId                             ReferenceId OPTIONAL,
      requestedAction                 [46]    IMPLICIT INTEGER{
                                                      resourceReport          (1),
                                                      resourceControl         (2),
                                                      cancel                  (3)},
      prefResourceReportFormat        [47]    IMPLICIT ResourceReportId OPTIONAL,
      resultSetWanted                 [48]    IMPLICIT BOOLEAN OPTIONAL,
      otherInfo                               OtherInformation OPTIONAL}



ResourceReportRequest ::= SEQUENCE{
      referenceId                             ReferenceId OPTIONAL,
      opId                            [210]   IMPLICIT ReferenceId OPTIONAL,
      prefResourceReportFormat        [49]    IMPLICIT ResourceReportId OPTIONAL,
      otherInfo                               OtherInformation OPTIONAL}
--
```

```
 ResourceReportResponse ::= SEQUENCE{
        referenceId                             ReferenceId OPTIONAL,
        resourceReportStatus           [50]     IMPLICIT INTEGER{
                                                        success             (0),
                                                        partial             (1),
                                                        failure-1           (2),
                                                        failure-2           (3),
                                                        failure-3           (4),
                                                        failure-4           (5),
                                                        failure-5           (6),
                                                        failure-6           (7)},
        resourceReport                 [51]     ResourceReport OPTIONAL,
        otherInfo                               OtherInformation OPTIONAL}
--
        ResourceReport          ::=     EXTERNAL
        ResourceReportId        ::=     OBJECT IDENTIFIER

--Scan APDUs
 ScanRequest ::= SEQUENCE{
        referenceId                                     ReferenceId OPTIONAL,
        databaseNames                           [3]     IMPLICIT SEQUENCE OF DatabaseName,
        attributeSet                                    AttributeSetId OPTIONAL,
        termListAndStartPoint                           AttributesPlusTerm,
        stepSize                                [5]     IMPLICIT INTEGER OPTIONAL,
        numberOfTermsRequested                  [6]     IMPLICIT INTEGER,
        preferredPositionInResponse             [7]     IMPLICIT INTEGER OPTIONAL,
        otherInfo                                       OtherInformation OPTIONAL}

 ScanResponse ::= SEQUENCE{
        referenceId                                     ReferenceId OPTIONAL,
        stepSize                                [3]     IMPLICIT INTEGER OPTIONAL,
        scanStatus                              [4]     IMPLICIT INTEGER {
                                                        success     (0),
                                                        partial-1   (1),
                                                        partial-2   (2),
                                                        partial-3   (3),
                                                        partial-4   (4),
                                                        partial-5   (5),
                                                        failure     (6) },
        numberOfEntriesReturned                 [5]     IMPLICIT INTEGER,
        positionOfTerm                          [6]     IMPLICIT INTEGER OPTIONAL,
        entries                                 [7]     IMPLICIT ListEntries  OPTIONAL,
        attributeSet                            [8]     IMPLICIT AttributeSetId OPTIONAL,
        otherInfo                                       OtherInformation OPTIONAL}

-- begin auxiliary definitions for Scan
 ListEntries ::= SEQUENCE{
        entries                        [1]      IMPLICIT SEQUENCE OF Entry OPTIONAL,
        nonsurrogateDiagnostics        [2]      IMPLICIT SEQUENCE OF DiagRec OPTIONAL
                -- At least one of entries and nonsurrogateDiagnostics must occur
                              }
```

```
  Entry  ::= CHOICE {
    termInfo                    [1]   IMPLICIT TermInfo,
    surrogateDiagnostic         [2]   DiagRec}
--
  TermInfo ::= SEQUENCE {
        term                          Term,
        displayTerm             [0]   IMPLICIT InternationalString OPTIONAL,
                                        -- Presence of displayTerm means that term is not considered by
                                        -- the target to be suitable for display, and displayTerm should
                                        -- instead be displayed. 'term' is the actual term in the term list;
                                        -- 'displayTerm' is for display purposes only, and is not an actual
                                        -- term in the term list.
        suggestedAttributes           AttributeList OPTIONAL,
        alternativeTerm         [4]   IMPLICIT SEQUENCE OF AttributesPlusTerm OPTIONAL,
        globalOccurrences       [2]   IMPLICIT INTEGER OPTIONAL,
        byAttributes            [3]   IMPLICIT OccurrenceByAttributes OPTIONAL,
        otherTermInfo                 OtherInformation OPTIONAL}

  OccurrenceByAttributes ::= SEQUENCE OF SEQUENCE{
        attributes          [1]   AttributeList,
        occurrences               CHOICE{
                                      global        [2] INTEGER,
                                      byDatabase    [3] IMPLICIT SEQUENCE OF SEQUENCE{
                                          db                  DatabaseName,
                                          num           [1]   IMPLICIT INTEGER OPTIONAL,
                                          otherDbInfo         OtherInformation OPTIONAL}} OPTIONAL,
        otherOccurInfo            OtherInformation OPTIONAL}
-- end auxiliary definitions for Scan

-- Sort APDUs
SortRequest  ::= SEQUENCE{
        referenceId                   ReferenceId OPTIONAL,
        inputResultSetNames     [3]   IMPLICIT SEQUENCE OF InternationalString,
        sortedResultSetName     [4]   IMPLICIT InternationalString,
        sortSequence            [5]   IMPLICIT SEQUENCE OF SortKeySpec,
                                        -- order of occurrence is from major to minor
        otherInfo                     OtherInformation OPTIONAL}

SortResponse  ::= SEQUENCE{
        referenceId                   ReferenceId OPTIONAL,
        sortStatus              [3]   IMPLICIT INTEGER{
                                          success       (0),
                                          partial-1     (1),
                                          failure       (2)},
        resultSetStatus         [4]   IMPLICIT INTEGER{
                                          empty         (1),
                                          interim       (2),
                                          unchanged     (3),
                                          none          (4)} OPTIONAL,
        diagnostics             [5]   IMPLICIT SEQUENCE OF DiagRec OPTIONAL,
        otherInfo                     OtherInformation OPTIONAL}
```

```
-- begin auxiliary definitions for Sort
        SortKeySpec ::= SEQUENCE{
          sortElement                          SortElement,
          sortRelation               [1]      IMPLICIT INTEGER{
                                                        ascending                   (0),
                                                        descending                  (1),
                                                        ascendingByFrequency        (3),
                                                        descendingByfrequency       (4)},
          caseSensitivity            [2]      IMPLICIT INTEGER{
                                                        caseSensitive               (0),
                                                        caseInsensitive             (1)},
      missingValueAction             [3]      CHOICE{
                                        abort                 [1] IMPLICIT NULL,
                                        null                  [2] IMPLICIT NULL,
                                                                  --supply a null value for missing value
                                      missingValueData    [3] IMPLICIT OCTET STRING} OPTIONAL}

        SortElement ::=       CHOICE{
          generic                    [1] SortKey,
          datbaseSpecific            [2] IMPLICIT SEQUENCE OF SEQUENCE{
                                        databaseName      DatabaseName,
                                        dbSort            SortKey}}

        SortKey ::= CHOICE{
                sortfield            [0]      IMPLICIT InternationalString,
                                              -- An element, element-group-tag, or alias supported by the target
                                              -- and denoting a set of elements associated with each record.
                elementSpec          [1]      IMPLICIT Specification,
        sortAttributes               [2]      IMPLICIT SEQUENCE{
                                                        id      AttributeSetId,
                                                        list    AttributeList}}
-- end auxiliary definitions for sort



-- Extended Service APDUs
  ExtendedServicesRequest  ::= SEQUENCE{
        referenceId                            ReferenceId OPTIONAL,
        function                     [3]      IMPLICIT INTEGER {
                                                        create        (1),
                                                        delete        (2),
                                                        modify        (3)},
        packageType                  [4]      IMPLICIT OBJECT IDENTIFIER,
        packageName                  [5]      IMPLICIT InternationalString OPTIONAL,
                                                        -- PackageName mandatory for 'modify' or 'delete'; optional for
                                                        -- 'create'. Following four parameters mandatory for 'create'; should
                                                        -- be included on 'modify' if being modified; not needed on 'delete'.
        userId                       [6]      IMPLICIT InternationalString OPTIONAL,
        retentionTime                [7]      IMPLICIT IntUnit OPTIONAL,
        permissions                  [8]      IMPLICIT Permissions OPTIONAL,
        description                  [9]      IMPLICIT InternationalString OPTIONAL,
```

```
-- (ExtendedServiceRequest APDU continued)
        taskSpecificParameters       [10]    IMPLICIT EXTERNAL OPTIONAL,
                                                     -- Mandatory for 'create'; included on 'modify' if specific
                                                     -- parameters being modified; not necessary on 'delete'. For the
                                                     -- 'EXTERNAL,' use OID of specific ES definition and select
                                                     --  CHOICE [1]: 'esRequest'.
        waitAction                   [11]    IMPLICIT INTEGER{
                                                     wait                 (1),
                                                     waitIfPossible       (2),
                                                     dontWait             (3),
                                                     dontReturnPackage    (4)},
        elements                                     ElementSetName OPTIONAL,
        otherInfo                                    OtherInformation OPTIONAL}
--

ExtendedServicesResponse ::= SEQUENCE{
        referenceId                                  ReferenceId OPTIONAL,
        operationStatus              [3]             IMPLICIT INTEGER{
                                                             done                 (1),
                                                             accepted             (2),
                                                             failure              (3)},
        diagnostics                  [4]             IMPLICIT SEQUENCE OF DiagRec OPTIONAL,
        taskPackage                  [5]             IMPLICIT EXTERNAL OPTIONAL,
                                                             -- Use OID: {Z39-50-recordSyntax (106)} and corresponding
                                                             -- syntax. For the EXTERNAL, 'taskSpecific,' within that
                                                             -- definition, use OID of the specific es, and choose [2],
                                                             -- 'taskPackage'.
        otherInfo                                    OtherInformation OPTIONAL}

 Permissions ::= SEQUENCE OF SEQUENCE{
        userId                 [1] IMPLICIT InternationalString,
        allowableFunctions     [2] IMPLICIT SEQUENCE OF INTEGER{
                                                     delete               (1),
                                                     modifyContents       (2),
                                                     modifyPermissions    (3),
                                                     present              (4),
                                                     invoke               (5)}}

Close ::= SEQUENCE{
        referenceId                                  ReferenceId OPTIONAL,  -- See 3.2.11.1.5.
        closeReason                                  CloseReason,
        diagnosticInformation              [3]       IMPLICIT InternationalString OPTIONAL,
        resourceReportFormat               [4]       IMPLICIT ResourceReportId OPTIONAL,
                                                             -- For use by origin only, and only on Close request;
                                                             -- origin requests target to include report in response.
        resourceReport                     [5]       ResourceReport OPTIONAL,
                                                             -- For use by target only, unilaterally on Close request;
                                                             -- on Close response may be unilateral or in response
                                                             -- to origin request.
        otherInfo                                    OtherInformation OPTIONAL}
```

```
CloseReason ::=    [211]  IMPLICIT INTEGER{
                          finished              (0),
                          shutdown              (1),
                          systemProblem         (2),
                          costLimit             (3),
                          resources             (4),
                          securityViolation     (5),
                          protocolError         (6),
                          lackOfActivity        (7),
                          peerAbort             (8),
                          unspecified           (9)}


-- Global auxiliary definitions
    ReferenceId          ::=        [2]    IMPLICIT OCTET STRING
    ResultSetId          ::=        [31]   IMPLICIT InternationalString
    ElementSetName       ::=        [103]  IMPLICIT InternationalString
    DatabaseName         ::=        [105]  IMPLICIT InternationalString
    AttributeSetId       ::=               OBJECT IDENTIFIER


-- OtherInformation
    OtherInformation    ::= [201] IMPLICIT SEQUENCE OF SEQUENCE{
        category                         [1]   IMPLICIT InfoCategory OPTIONAL,
        information           CHOICE{
            characterInfo                [2]    IMPLICIT InternationalString,
            binaryInfo                   [3]    IMPLICIT OCTET STRING,
            externallyDefinedInfo        [4]    IMPLICIT EXTERNAL,
            oid                          [5]    IMPLICIT OBJECT IDENTIFIER}}
--
    InfoCategory ::= SEQUENCE{
            categoryTypeId    [1]    IMPLICIT OBJECT IDENTIFIER OPTIONAL,
            categoryValue     [2]    IMPLICIT INTEGER}


-- Units
    -- IntUnit is used when value and unit are supplied together. Unit, alone, is used when just
    -- specifying a unit (without a value).  For example, IntUnit is used in Term, in an RPNQuery, or
    -- it can be the datatype of an element within a retrieval record. Unit (alone) would be used in an
    -- element request, when requesting data be returned according to a particular unit.

    IntUnit ::= SEQUENCE{
        value         [1] IMPLICIT INTEGER,
        unitUsed      [2] IMPLICIT Unit}
--
    Unit ::= SEQUENCE{
        unitSystem        [1] InternationalString OPTIONAL,          -- e.g. 'SI'
        unitType          [2] StringOrNumeric OPTIONAL,              -- e.g. 'mass'
        unit              [3] StringOrNumeric OPTIONAL,              -- e.g. 'kilograms'
        scaleFactor       [4] IMPLICIT INTEGER OPTIONAL              -- e.g. 9 means 10**9
                    }
```

```
--CharacterString
        InternationalString ::= GeneralString
                -- When version 2 is in force, this collapses to VisibleString. That is, only characters in the
                -- visibleString repertoire may be used. (Datatype compatibility with version 2 is not affected,
                -- because references are IMPLICIT.)  When version 3 is in force, the semantics of the
                -- GeneralString content may be altered by negotiation during initialization. If no such
                -- negotiation is in effect, then GeneralString semantics are in force.

StringOrNumeric ::= CHOICE{
        string          [1] IMPLICIT InternationalString,
        numeric         [2] IMPLICIT INTEGER}

END -- IR DEFINITIONS



DiagnosticFormatDiag1
{Z39-50-diagnosticFormat diag-1 (2)} DEFINITIONS ::=
BEGIN
IMPORTS Term, Specification, AttributeList, SortElement, DatabaseName,
DefaultDiagFormat, InternationalString FROM Z39-50-APDU-1995;


DiagnosticFormat ::=  SEQUENCE OF SEQUENCE{
   diagnostic    [1] CHOICE{
                        defaultDiagRec          [1]     IMPLICIT DefaultDiagFormat,
                        explicitDiagnostic      [2]     DiagFormat} OPTIONAL,
   message  [2] IMPLICIT InternationalString OPTIONAL}


DiagFormat ::= CHOICE{

tooMany     [1000]    IMPLICIT SEQUENCE{
                          tooManyWhat  [1]     IMPLICIT INTEGER{
                                                  argumentWords           (1),
                                                  truncatedWords          (2),
                                                  booleanOperators        (3),
                                                  incompleteSubfields     (4),
                                                  characters              (5),
                                                  recordsRetrieved        (6),
                                                  dataBasesSpecified      (7),
                                                  resultSetsCreated       (8),
                                                  indexTermsProcessed     (9)},
                          max          [2]     IMPLICIT INTEGER OPTIONAL},
```

```
badSpec     [1001]   IMPLICIT SEQUENCE{  -- element set name or specification
                        spec          [1] IMPLICIT Specification, -- esn or element spec not supported
                        db            [2] IMPLICIT DatabaseName OPTIONAL,
                                          -- if db specified, above spec not supported for db; otherwise,
                                          -- spec not supported period.
                        goodOnes      [3] IMPLICIT SEQUENCE OF Specification OPTIONAL
                                          -- target supplies ones that are supported
                                          },


dbUnavail   [1002]   IMPLICIT SEQUENCE{ -- database unavailable
                        db    [1] IMPLICIT DatabaseName,
                        why   [2] IMPLICIT SEQUENCE{
                                 reasonCode [1] IMPLICIT INTEGER{
                                                    doesNotExist      (0),
                                                    existsButUnavail  (1),
                                                    locked            (2),
                                                    accessDenied      (3)} OPTIONAL,
                                 message    [2] IMPLICIT InternationalString OPTIONAL}},


unSupOp     [1003]   IMPLICIT INTEGER{  -- unsupported operator
                        and      (0),
                        or       (1),
                        and-not  (2),
                        prox     (3)},


attribute   [1004]   IMPLICIT SEQUENCE{
                           -- Applies for unsupported attribute set, attribute type,
                           -- attribute value, or term (for a given attribute type or value).

                        id       [1]   IMPLICIT OBJECT IDENTIFIER,
                           -- if only "id" occurs, then attribute set is not supported
                        type     [2]   IMPLICIT INTEGER OPTIONAL,
                           -- must occur if value occurs.
                        value    [3]   IMPLICIT INTEGER OPTIONAL,
                           -- if omitted, and Type occurs, then Type is what is unsupported
                        term     [4]   Term OPTIONAL
                           -- If occurs, term is illegal or not supported, for attribute value,
                           -- if value occurs; otherwise, for type.
                                          },


attCombo    [1005]   IMPLICIT SEQUENCE{ -- attribute combination not supported
                     unsupportedCombination     [1] IMPLICIT AttributeList,
                     recommendedAlternatives    [2] IMPLICIT SEQUENCE OF AttributeList OPTIONAL},
```

```
term        [1006]  IMPLICIT SEQUENCE{
                        problem     [1] IMPLICIT INTEGER{
                                        codedValue  (1),
                                        unparsable  (2),
                                        tooShort    (3),
                                        type        (4)} OPTIONAL,
                        term        [2] Term},


proximity   [1007] CHOICE{          -- proximity diagnostics:
                resultSets  [1]  IMPLICIT NULL,               -- proximity between sets not supported
                badSet      [2]  IMPLICIT InternationalString,  -- bad result set specified
                relation    [3]  IMPLICIT INTEGER,            -- 1 to 6 ; relation not supported
                unit        [4]  IMPLICIT INTEGER,            -- unsupported unit code
                distance    [5]  IMPLICIT INTEGER,            -- unsupported distance
                attributes  [6]  AttributeList,               --   proximity not supported with specified
                                                              -- attribute combination
                ordered     [7]  IMPLICIT NULL,               -- ordered flag not supported
                exclusion   [8]  IMPLICIT NULL                -- exclusion flag not supported
                        },


scan     [1008] CHOICE{         -- scan diagnostics:
                nonZeroStepSize   [0] IMPLICIT NULL,    -- only zero step size supported
                specifiedStepSize [1] IMPLICIT NULL,    -- specified step size not supported
                termList1         [3] IMPLICIT NULL,    -- term list not supported (no alternative supplied)
                termList2         [4] IMPLICIT SEQUENCE OF AttributeList,
                                                        -- term list not supported (alternatives supplied)
                posInResponse     [5] IMPLICIT INTEGER{   --value of positionInResponse not supported
                                        mustBeOne           (1),
                                        mustBePositive      (2),
                                        mustBeNonNegative   (3),
                                        other               (4)},
                resources         [6] IMPLICIT NULL,    -- resources exhausted looking for satisfying terms
                endOfList         [7] IMPLICIT NULL     -- beginning or end of term list
                    },


sort        [1009] CHOICE{
                sequence      [0]   IMPLICIT NULL,          -- cannot sort according to sequence
                noRsName      [1]   IMPLICIT NULL,          -- no result set name supplied
                tooMany       [2]   IMPLICIT INTEGER,       --   Too many input result sets,
                                                            --  maximum supplied.
                incompatible  [3]   IMPLICIT NULL,          -- records with different formats
                                                            -- not compatible for sorting
                generic       [4]   IMPLICIT NULL,          -- generic sort not supported
                                                            -- (db specific only)
                dbSpecific    [5]   IMPLICIT NULL,          -- db specific sort not supported
```

```
            sortElement      [6]    SortElement,
            key              [7]    IMPLICIT INTEGER{
                                        tooMany     (1),        -- too many sort keys
                                        duplicate   (2)},       -- duplicate sort keys
            action           [8]    IMPLICIT NULL,     -- unsupported missing data action
            illegal          [9]    IMPLICIT INTEGER{
                                        relation  (1),   -- illegal sort relation
                                        case      (2),   -- illegal case value
                                        action    (3),   -- illegal missing data action
                                        sort      (4)},  -- illegal sort
         inputTooLarge       [10]   IMPLICIT SEQUENCE OF InternationalString,
                                                 --   one or more of the input result sets too large to sort
        aggregateTooLarge    [11]   IMPLICIT NULL --  aggregate result set too large
                                  },

segmentation   [1010]   CHOICE{
                        segmentCount  [0]       IMPLICIT NULL,
                                                    -- Cannot guarantee record will fit within max segments. Target
                                                    -- suggests that origin try again to retrieve record, without
                                                    -- including max-segment-count.
                        segmentSize   [1]       IMPLICIT INTEGER
                                                -- record cannot be segmented into fragments such that the largest
                                                -- will fit within max segment size specified. Target supplies (in
                                                -- bytes) the smallest acceptable value of Max-segment-size to
                                                -- retrieve the record.
                                  },


extServices  [1011] CHOICE{
                        req              [1] IMPLICIT INTEGER{    -- bad request
                                            nameInUse      (1),    -- package name already in use
                                            noSuchName     (2),    -- no such package, on modify/delete
                                            quota          (3),    -- quota exceeded
                                            type           (4)},   -- extended service type not supported
                        permission   [2] IMPLICIT INTEGER{ -- permission denied on ES, because:
                                            id             (1),        --   id not authorized, or
                                            modifyDelete   (2)},       -- cannot modify or delete
                        immediate    [3] IMPLICIT INTEGER{ -- immediate execution:
                                            failed         (1),        -- failed,
                                            service        (2),        -- not supported for this service, or
                                            parameters     (3)         -- for these parameters.
                                                  }},


accessCtrl       [1012] CHOICE{
                        noUser       [1] IMPLICIT NULL,         -- no user to display challenge to
                        refused      [2] IMPLICIT NULL,         -- access control information refused by user
                        simple       [3] IMPLICIT NULL,         -- only simple form supported (target used
                                                                -- externally defined)
                        oid          [4] IMPLICIT SEQUENCE OF OBJECT IDENTIFIER,
                                                                -- oid not supported (origin supplies alternative
```

```
                                                    -- suggested oids)
                alternative   [5] IMPLICIT SEQUENCE OF OBJECT IDENTIFIER,
                                                    -- origin insists that target use an alternative
                                                    -- challenge for this data (e.g. stronger
                                                    -- authentication or stronger Access control). The
                                                    -- origin supplies suggested alternative oids.
                pwdInv        [6] IMPLICIT NULL,    -- password invalid
                pwdExp        [7] IMPLICIT NULL     -- password expired
                    },


recordSyntax   [1013] IMPLICIT SEQUENCE{ -- record cannot be transferred in requested syntax
               unsupportedSyntax       [1] IMPLICIT OBJECT IDENTIFIER,
               suggestedAlternatives   [2] IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL}
}

END
```

REC.1 Explain Record Syntax

```
RecordSyntax-explain
{Z39-50-recordSyntax explain (100)} DEFINITIONS ::=

BEGIN
IMPORTS AttributeSetId, Term, OtherInformation, DatabaseName, ElementSetName, IntUnit, Unit,
 StringOrNumeric, Specification, InternationalString, AttributeList, AttributeElement FROM Z39-50-APDU-1995;

Explain-Record ::= CHOICE{
    -- Each of these may be used as search term  when Use attribute is 'explain-category'.
  targetInfo             [0]     IMPLICIT TargetInfo,
  databaseInfo           [1]     IMPLICIT DatabaseInfo,
  schemaInfo             [2]     IMPLICIT SchemaInfo,
  tagSetInfo             [3]     IMPLICIT TagSetInfo,
  recordSyntaxInfo       [4]     IMPLICIT RecordSyntaxInfo,
  attributeSetInfo       [5]     IMPLICIT AttributeSetInfo,
  termListInfo           [6]     IMPLICIT TermListInfo,
  extendedServicesInfo   [7]     IMPLICIT ExtendedServicesInfo,
  attributeDetails       [8]     IMPLICIT AttributeDetails,
  termListDetails        [9]     IMPLICIT TermListDetails,
  elementSetDetails      [10]    IMPLICIT ElementSetDetails,
  retrievalRecordDetails [11]    IMPLICIT RetrievalRecordDetails,
  sortDetails            [12]    IMPLICIT SortDetails,
  processing             [13]    IMPLICIT ProcessingInformation,
  variants               [14]    IMPLICIT VariantSetInfo,
  units                  [15]    IMPLICIT UnitInfo,
  categoryList           [100]   IMPLICIT CategoryList}
```

```
-- Element set name 'B' (brief)  retrieves:
--    -  'commonInfo' (except for otherInfo within commonInfo)
--    -  key elements
--    -  other elements designated as 'non-key brief elements'
-- Esn 'description' retrieves brief elements as well as 'description', and specific additional descriptive
-- elements if designated.
-- Element set name 'F' (full) retrieves all of the above, as well as those designated as "non-brief elements". Some
-- elements designated as OPTIONAL may be mandatory in full records, and are so identified. (Note that all elements
-- that are not part of the brief element set must be designated as OPTIONAL in the ASN.1, otherwise it would be
-- illegal to omit them.)
-- Other esns are defined (below) as needed.

-- - - - - - - - - - - - -   Info Records
  -- Info records are mainly for software consumption
  -- They describe individual entities within the target system:
  --   The target itself
  --   Individual databases
  --   Schemas
  --   Tag sets
  --   Record syntaxes
  --   Attribute sets
  --   Term lists
  --   Extended services
 --        The information about each Schema, Tag Set, Record Syntax and Attribute Set should
  -- match the universal definitions of these items. The only exception is that a target may omit any
  -- items it doesn't support, for example the description of the bib-1 attribute set may omit attributes
  -- that the target does not support under any circumstances.
  --     Databases that may be searched together can be listed in the dbCominations element of the TargetInfo record.

TargetInfo ::= SEQUENCE {
  commonInfo            [0]     IMPLICIT CommonInfo OPTIONAL,
     -- Key elements follow:
  name                  [1]     IMPLICIT InternationalString,
     -- Non-key brief elements follow:
  recent-news           [2]     IMPLICIT HumanString OPTIONAL,
  icon                  [3]     IMPLICIT IconObject OPTIONAL,
  namedResultSets       [4]     IMPLICIT BOOLEAN,
  multipleDBsearch      [5]     IMPLICIT BOOLEAN,
  maxResultSets         [6]     IMPLICIT INTEGER OPTIONAL,
  maxResultSize         [7]     IMPLICIT INTEGER OPTIONAL,
  maxTerms              [8]     IMPLICIT INTEGER OPTIONAL,
  timeoutInterval       [9]     IMPLICIT IntUnit OPTIONAL,
  welcomeMessage        [10]    IMPLICIT HumanString OPTIONAL,
     -- non-brief elements follow:
     -- 'description' esn retrieves the following two (as well as brief):
  contactInfo           [11]    IMPLICIT ContactInfo OPTIONAL,
  description           [12]    IMPLICIT HumanString OPTIONAL,
  nicknames             [13]    IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
  usage-restrictions    [14]    IMPLICIT HumanString OPTIONAL,
  paymentAddr           [15]    IMPLICIT HumanString OPTIONAL,
  hours                 [16]    IMPLICIT HumanString OPTIONAL,
```

```
  dbCombinations          [17]    IMPLICIT SEQUENCE OF DatabaseList OPTIONAL,
  addresses               [18]    IMPLICIT SEQUENCE OF NetworkAddress OPTIONAL,
  languages               [101]   IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
                                      -- Languages supported for message strings.  Each is a three-character
                                      -- language code from Z39.53-1994.
-- characterSets          [102]  this tag reserved for "character sets supported for name and message strings".
  -- commonAccessInfo elements list objects the target supports. All objects listed in
  -- AccessInfo for any individual database should also be listed here.
  commonAccessInfo        [19]    IMPLICIT AccessInfo OPTIONAL}


DatabaseInfo ::= SEQUENCE {
                -- A target may provide "virtual databases" that are combinations of individual database. These
                -- databases are indicated by the presence of subDbs in the combination database's DatabaseDescription.
  commonInfo                [0]     IMPLICIT CommonInfo OPTIONAL,
   -- Key elements follow:
  name                      [1]     IMPLICIT DatabaseName,
   -- Non-key brief elements follow:
  explainDatabase           [2]     IMPLICIT NULL OPTIONAL,
                                        -- If present, this database is the Explain database, or an Explain database
                                        -- for a different server, possibly on a different host. The means by which
                                        -- that server may be accessed is not addressed by this standard. One
                                        -- suggested possibility is an implementor agreement whereby the
                                        -- database name is a url which may be used to connect to the server.
  nicknames                 [3]     IMPLICIT SEQUENCE OF DatabaseName OPTIONAL,
  icon                      [4]     IMPLICIT IconObject OPTIONAL,
  user-fee                  [5]     IMPLICIT BOOLEAN,
  available                 [6]     IMPLICIT BOOLEAN,
  titleString               [7]     IMPLICIT HumanString OPTIONAL,
   -- Non-brief elements follow:
  keywords                  [8]     IMPLICIT SEQUENCE OF HumanString OPTIONAL,
  description               [9]     IMPLICIT HumanString OPTIONAL,
  associatedDbs             [10]    IMPLICIT DatabaseList OPTIONAL,
                                        -- databases that may be searched in combination with this one

  subDbs                    [11]    IMPLICIT DatabaseList OPTIONAL,
                                        -- When present, this database is a composite representing the combined
                                        -- databases 'subDbs'. The individual subDbs are also available.
  disclaimers               [12]    IMPLICIT HumanString OPTIONAL,
  news                      [13]    IMPLICIT HumanString OPTIONAL,
  recordCount               [14]    CHOICE {
                                        actualNumber     [0] IMPLICIT INTEGER,
                                        approxNumber     [1] IMPLICIT INTEGER} OPTIONAL,
  defaultOrder              [15]    IMPLICIT HumanString OPTIONAL,
  avRecordSize              [16]    IMPLICIT INTEGER OPTIONAL,
  maxRecordSize             [17]    IMPLICIT INTEGER OPTIONAL,
  hours                     [18]    IMPLICIT HumanString OPTIONAL,
  bestTime                  [19]    IMPLICIT HumanString OPTIONAL,
  lastUpdate                [20]    IMPLICIT GeneralizedTime OPTIONAL,
  updateInterval            [21]    IMPLICIT IntUnit OPTIONAL,
  coverage                  [22]    IMPLICIT HumanString OPTIONAL,
```

```
  proprietary                  [23]     IMPLICIT BOOLEAN OPTIONAL, -- mandatory in full record
  copyrightText                [24]     IMPLICIT HumanString OPTIONAL,
  copyrightNotice              [25]     IMPLICIT HumanString OPTIONAL,
  producerContactInfo          [26]     IMPLICIT ContactInfo OPTIONAL,
  supplierContactInfo          [27]     IMPLICIT ContactInfo OPTIONAL,
  submissionContactInfo        [28]     IMPLICIT ContactInfo OPTIONAL,
    -- accessInfo lists items connected with the database. All listed items should be in the target's AccessInfo.
  accessInfo                   [29]     IMPLICIT AccessInfo OPTIONAL}


SchemaInfo ::= SEQUENCE {
  commonInfo          [0]  IMPLICIT CommonInfo OPTIONAL,
    -- Key elements follow:
  schema              [1]  IMPLICIT OBJECT IDENTIFIER,
    -- Non-key brief elements follow:
  name                [2]  IMPLICIT InternationalString,
    -- Non-brief elements follow:
  description         [3]  IMPLICIT HumanString OPTIONAL,
  tagTypeMapping      [4]  IMPLICIT SEQUENCE OF SEQUENCE {
                               tagType            [0] IMPLICIT INTEGER,
                               tagSet             [1] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
                                   -- If tagSet is omitted, then this tagType is for a tagSet locally defined
                                   -- within the schema that cannot be referenced by another schema.
                               defaultTagType     [2] IMPLICIT NULL OPTIONAL
                                                      } OPTIONAL,
  recordStructure     [5]  IMPLICIT SEQUENCE OF ElementInfo OPTIONAL}


    --  ElementInfo referenced in SchemaInfo and RecordSyntaxInfo
        ElementInfo ::= SEQUENCE {
            elementName      [1] IMPLICIT InternationalString,
            elementTagPath   [2] IMPLICIT Path,
            dataType         [3] ElementDataType OPTIONAL, -- If omitted, not specified.
            required         [4] IMPLICIT BOOLEAN,
            repeatable       [5] IMPLICIT BOOLEAN,
            description      [6] IMPLICIT HumanString OPTIONAL}


    -- Path is referenced by ElementInfo as well as PerElementDetails
                Path ::= SEQUENCE OF SEQUENCE{
                tagType      [1] IMPLICIT INTEGER,
                tagValue     [2] StringOrNumeric}
            ElementDataType ::= CHOICE{
                primitive    [0] IMPLICIT PrimitiveDataType,
                structured   [1] IMPLICIT SEQUENCE OF ElementInfo}
            PrimitiveDataType ::= INTEGER{
                octetString          (0),
                numeric              (1),
                date                 (2),
                external             (3),
                string               (4),
```

```
                    trueOrFalse         (5),
                    oid                 (6),
                    intUnit             (7),
                    empty               (8),
                    noneOfTheAbove      (100) -- see 'description'
                              }


TagSetInfo ::= SEQUENCE {
  commonInfo    [0]  IMPLICIT CommonInfo OPTIONAL,
     -- Key elements follow:
  tagSet        [1]  IMPLICIT OBJECT IDENTIFIER,
     -- non-key brief elements follow:
  name          [2]  IMPLICIT InternationalString,
     -- non-brief elements follow:
  description   [3]  IMPLICIT HumanString OPTIONAL,
  elements      [4]  IMPLICIT SEQUENCE OF SEQUENCE {
                          elementname      [1]  IMPLICIT InternationalString,
                          nicknames        [2]  IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
                          elementTag       [3]  StringOrNumeric,
                          description      [4]  IMPLICIT HumanString OPTIONAL,
                          dataType         [5]  PrimitiveDataType OPTIONAL,
                               -- If the data type is expected to be structured, that is described in the schema info,
                               -- and datatype is omitted here.
                          otherTagInfo          OtherInformation OPTIONAL} OPTIONAL}

RecordSyntaxInfo ::= SEQUENCE {
  commonInfo         [0]  IMPLICIT CommonInfo OPTIONAL,
     -- Key elements follow:
  recordSyntax       [1]  IMPLICIT OBJECT IDENTIFIER,
     -- Non-key brief elements follow:
  name               [2]  IMPLICIT InternationalString,
     -- non-brief elements follow:
  transferSyntaxes   [3]  IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
  description        [4]  IMPLICIT HumanString OPTIONAL,
  asn1Module         [5]  IMPLICIT InternationalString OPTIONAL,
  abstractStructure  [6]  IMPLICIT SEQUENCE OF ElementInfo OPTIONAL
                               -- Omitting abstractStructure only means target isn't using
                               -- Explain to describe the structure, not that there is no structure.
                          }

AttributeSetInfo ::= SEQUENCE {
  commonInfo    [0]  IMPLICIT CommonInfo OPTIONAL,
     -- Key elements follow:
  attributeSet  [1]  IMPLICIT AttributeSetId,
     -- non-key brief elements follow:
  name          [2]  IMPLICIT InternationalString,
     -- non-brief elements follow:
  attributes    [3]  IMPLICIT SEQUENCE OF AttributeType OPTIONAL,
                          -- mandatory in full record
  description   [4] IMPLICIT HumanString OPTIONAL}
```

```
-- AttributeType referenced in AttributeSetInfo
    AttributeType ::= SEQUENCE {
            name              [0] IMPLICIT InternationalString OPTIONAL,
            description       [1] IMPLICIT HumanString OPTIONAL,
            attributeType     [2] IMPLICIT INTEGER,
            attributeValues   [3] IMPLICIT SEQUENCE OF AttributeDescription}
        AttributeDescription ::= SEQUENCE {
                name                  [0] IMPLICIT InternationalString OPTIONAL,
                description           [1] IMPLICIT HumanString OPTIONAL,
                attributeValue        [2] StringOrNumeric,
                equivalentAttributes  [3]  IMPLICIT SEQUENCE OF StringOrNumeric OPTIONAL
                                      -- each is an occurrence of 'attributeValue' from AttributeDescription for a
                                      -- different attribute. Equivalences listed here should be derived from the
                                      -- attribute set definition, not from a particular server's behavior.
                                          }


TermListInfo ::= SEQUENCE{
  commonInfo          [0]  IMPLICIT CommonInfo OPTIONAL,
    -- Key elements follow:
  databaseName  [1]  IMPLICIT DatabaseName,
    -- Non-key brief elements follow:
  termLists             [2] IMPLICIT SEQUENCE OF SEQUENCE{
                            name          [1] IMPLICIT InternationalString,
                            title         [2] IMPLICIT HumanString OPTIONAL,
                                                  -- Title is for users to see and can differ by language. Name, on the
                                                  -- other hand is typically a short string not necessarily meant to be
                                                  -- human-readable, and not variable by language.
                            searchCost    [3]   IMPLICIT INTEGER {
                                          optimized     (0),        -- The attribute (or combination) associated
                                                                    -- with this    list will do fast searches.
                                          normal        (1),        -- The attribute (combination) will work as
                                                                    -- expected. So there's probably an index for the
                                                                    -- attribute (combination) or some similar
                                                                    -- mechanism.
                                          expensive     (2),        -- Can use the attribute (combination), but it
                                                                    -- might not provide satisfactory results.
                                                                    -- Probably there is no index, or post-
                                                                    -- processing of records is required.
                                          filter        (3)         -- can't search with this attribute (combination) alone.
                                                        } OPTIONAL,
                            scanable      [4]  IMPLICIT BOOLEAN, -- 'true' means this list can be scanned.
                            broader       [5] IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
                            narrower      [6] IMPLICIT SEQUENCE OF InternationalString OPTIONAL
                                              -- broader and narrower list alternative term lists related to this one.
                                              -- The term lists so listed should also be in this termLists structure.
                                                          }
    -- no non-brief elements
                            }
```

```
ExtendedServicesInfo ::= SEQUENCE {
  commonInfo              [0]  IMPLICIT CommonInfo OPTIONAL,
     -- Key elements follow:
  type                    [1]  IMPLICIT OBJECT IDENTIFIER,
     -- Non-key brief elements follow:
  name                    [2]  IMPLICIT InternationalString OPTIONAL,
                                    -- should be supplied if privateType is 'true'
  privateType             [3]  IMPLICIT BOOLEAN,
  restrictionsApply       [5]  IMPLICIT BOOLEAN,  -- if 'true' see 'description'
  feeApply                [6]  IMPLICIT BOOLEAN,  -- if 'true' see 'description'
  available               [7]  IMPLICIT BOOLEAN,
  retentionSupported      [8]  IMPLICIT BOOLEAN,
  waitAction              [9]  IMPLICIT INTEGER{
                                    waitSupported        (1),
                                    waitAlways           (2),
                                    waitNotSupported     (3),
                                    depends              (4),
                                    notSaying            (5)},
  -- non-brief elements follow:
        -- To get brief plus 'description' use esn 'description'
  description             [10]      IMPLICIT HumanString OPTIONAL,
        -- to get above elements and 'specificExplain' use esn 'specificExplain'
  specificExplain         [11]      IMPLICIT EXTERNAL OPTIONAL,
                                         -- Use oid of specific ES, and select choice [3] 'explain'. Format
                                         -- to be developed in conjunction with the specific ES definition.
        -- to get all elements except 'specificExplain', use esn 'asn'
  esASN                   [12]      IMPLICIT InternationalString OPTIONAL -- the ASN.1 for this ES
                                    }

-- - - - - - - - - - - - Detail records
 -- The detail records describe relationships among entities supported by the target. RetrievalRecordDetails describes
 -- the way that schema elements are mapped into record elements. This mapping may be different for each
 -- combination of database, schema, record syntax. The per-element details describe the default mapping.
 -- Origin-request re-tagging can change that mapping. When multiple databases are listed in a databaseNames
 -- element, the record applies equally to all of the listed databases.  This is unrelated to searching the databases
 -- together. AttributeDetails describes how databases can be searched. Each supported attribute is listed, and the
 -- allowable combinations can be described.


AttributeDetails ::= SEQUENCE {
  commonInfo                   [0]  IMPLICIT CommonInfo OPTIONAL,
     -- Key elements follow:
  databaseName                 [1]  IMPLICIT DatabaseName,
     -- Non-brief elements follow:
  attributesBySet              [2]  IMPLICIT SEQUENCE OF AttributeSetDetails OPTIONAL,
                                         -- mandatory in full record
  attributeCombinations        [3]  IMPLICIT AttributeCombinations OPTIONAL}
```

```
-- AttributeSetDetails referenced by AttributeDetails
    AttributeSetDetails ::= SEQUENCE {
        attributeSet            [0] IMPLICIT AttributeSetId,
        attributesByType        [1] IMPLICIT SEQUENCE OF AttributeTypeDetails }
      AttributeTypeDetails ::= SEQUENCE {
            attributeType           [0] IMPLICIT INTEGER,
            defaultIfOmitted        [1] IMPLICIT OmittedAttributeInterpretation OPTIONAL,
            attributeValues         [2] IMPLICIT SEQUENCE OF AttributeValue OPTIONAL }
                                        -- If no attributeValues are supplied, all values of this type are fully
                                        -- supported, and the descriptions in AttributeSetInfo are adequate.

        OmittedAttributeInterpretation ::= SEQUENCE {
            defaultValue            [0] StringOrNumeric OPTIONAL,
                                        -- A default value is listed if that's how the server works
            defaultDescription      [1] IMPLICIT HumanString OPTIONAL }
                                        -- The human-readable description should generally be provided.
                                        -- It is legal for both default elements to be missing, which
                                        -- means that the target will allow the attribute type to be
                                        -- omitted, but isn't saying what it will do.

        AttributeValue ::= SEQUENCE {
          value                 [0]  StringOrNumeric,
          description           [1]  IMPLICIT HumanString OPTIONAL,
          subAttributes         [2]  IMPLICIT SEQUENCE OF StringOrNumeric OPTIONAL,
          superAttributes       [3]  IMPLICIT SEQUENCE OF StringOrNumeric OPTIONAL,
          partialSupport        [4]  IMPLICIT NULL OPTIONAL }
                -- partialSupport indicates that an attributeValue is accepted, but may not be processed in the
                -- "expected" way. One important reason for this is composite databases: in this case partialSupport
                -- may indicate that only some of the subDbs support the attribute, and others ignore it.



TermListDetails ::= SEQUENCE{  -- one for each termList in TermListInfo
  commonInfo          [0]  IMPLICIT CommonInfo OPTIONAL,
    -- Key elements follow:
  termListName        [1]  IMPLICIT InternationalString,
    -- Non-key elements (all non-brief) follow:
  description         [2]  IMPLICIT HumanString OPTIONAL,
  attributes          [3]  IMPLICIT AttributeCombinations OPTIONAL,
                           -- Pattern for attributes that hit this list. Mandatory in full record
  scanInfo            [4]  IMPLICIT SEQUENCE {
                               maxStepSize          [0] IMPLICIT INTEGER OPTIONAL,
                               collatingSequence    [1] IMPLICIT HumanString OPTIONAL,
                               increasing           [2] IMPLICIT BOOLEAN OPTIONAL} OPTIONAL,
                           -- Occurs only if list is scanable. If list is scanable and if scanInfo is omitted,
                           -- target doesn't consider these important.
  estNumberTerms      [5]  IMPLICIT INTEGER OPTIONAL,
  sampleTerms         [6]  IMPLICIT SEQUENCE OF Term OPTIONAL}
```

```
ElementSetDetails ::= SEQUENCE {
    -- ElementSetDetails describes the way that database records are mapped to record elements. This
    -- mapping may be different for each combination of database name and element set. The database record
    -- description is a schema, which may be private to the target. The schema's abstract record structure
    -- and tag sets provide the vocabulary for discussing record content; their presence in the Explain
    -- database does not imply support for complex retrieval specification.
  commonInfo            [0]  IMPLICIT CommonInfo OPTIONAL,
    -- Key elements follow:
  databaseName          [1]  IMPLICIT DatabaseName,
  elementSetName        [2]  IMPLICIT ElementSetName,
  recordSyntax          [3]  IMPLICIT OBJECT IDENTIFIER,
    -- Non-key Brief elements follow:
  schema                [4]  IMPLICIT OBJECT IDENTIFIER,
    -- Non-brief elements follow:
  description           [5]  IMPLICIT HumanString OPTIONAL,
  detailsPerElement     [6]  IMPLICIT SEQUENCE OF PerElementDetails OPTIONAL -- mandatory in full record
                             }


RetrievalRecordDetails ::= SEQUENCE {
  commonInfo            [0]  IMPLICIT CommonInfo OPTIONAL,
      -- Key elements follow:
  databaseName          [1]  IMPLICIT DatabaseName,
  schema                [2]  IMPLICIT OBJECT IDENTIFIER,
  recordSyntax          [3]  IMPLICIT OBJECT IDENTIFIER,
    -- Non-brief elements follow:
  description           [4]  IMPLICIT HumanString OPTIONAL,
  detailsPerElement     [5]  IMPLICIT SEQUENCE OF PerElementDetails OPTIONAL
                                  -- mandatory in full record
                        }

-- PerElementDetails is referenced in RetrievalRecordDetails and ElementSetDetails.
    PerElementDetails ::= SEQUENCE {
        name            [0]     IMPLICIT InternationalString OPTIONAL,
                                    -- If the name is omitted, the record syntax's name for this element
                                    -- is appropriate.
        recordTag       [1]     IMPLICIT RecordTag OPTIONAL,
                                    -- The record tag may be omitted if tags are inappropriate for the record
                                    -- syntax, or if the origin can be expected to know it for some other reason.
        schemaTags      [2]     IMPLICIT SEQUENCE OF Path OPTIONAL,
                                    -- The information from the listed schema elements is combined
                                    -- in some way to produce the data sent in the listed record tag. The
                                    -- 'contents' element below may describe the logic used.
        maxSize         [3]     IMPLICIT INTEGER OPTIONAL,
        minSize         [4]     IMPLICIT INTEGER OPTIONAL,
        avgSize         [5]     IMPLICIT INTEGER OPTIONAL,
        fixedSize       [6]     IMPLICIT INTEGER OPTIONAL,
        repeatable      [8]     IMPLICIT BOOLEAN,
        required        [9]     IMPLICIT BOOLEAN,
                                    -- 'required' really means that target will always supply the element.
        description     [12]    IMPLICIT HumanString OPTIONAL,
```

```
        contents          [13]    IMPLICIT HumanString OPTIONAL,
        billingInfo       [14]    IMPLICIT HumanString OPTIONAL,
        restrictions      [15]    IMPLICIT HumanString OPTIONAL,
        alternateNames    [16]    IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
        genericNames      [17]    IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
        searchAccess      [18]    IMPLICIT AttributeCombinations OPTIONAL }
            -- RecordTag referenced in PerElementDetails above
                RecordTag ::= SEQUENCE {
                    qualifier     [0] StringOrNumeric OPTIONAL,
                                            -- E.g. tag set for GRS-1
                    tagValue      [1] StringOrNumeric}

SortDetails ::= SEQUENCE {
  commonInfo          [0]  IMPLICIT CommonInfo OPTIONAL,
    -- Key elements follow:
  databaseName        [1]  IMPLICIT DatabaseName,
    -- No non-key brief elements
    -- Non-brief elements follow:
  sortKeys            [2]  IMPLICIT SEQUENCE OF SortKeyDetails OPTIONAL
                                  -- mandatory in full record
                      }
    SortKeyDetails ::= SEQUENCE {
        description                [0]  IMPLICIT HumanString OPTIONAL,
        elementSpecifications      [1]  IMPLICIT SEQUENCE OF Specification OPTIONAL,
                                            -- each specification is a way of specifying this same sort key
        attributeSpecifications    [2]  IMPLICIT AttributeCombinations OPTIONAL,
                                            -- each combination is a way of specifying this same sort key
        sortType                   [3]  CHOICE {
                                            character      [0]  IMPLICIT NULL,
                                            numeric        [1]  IMPLICIT NULL,
                                            structured     [2]  IMPLICIT HumanString} OPTIONAL,
        caseSensitivity            [4]  IMPLICIT INTEGER {
                                            always      (0), -- always case-sensitive
                                            never       (1), -- never case-sensitive
                                            default-yes (2), -- case-sensitivity is as specified on request, and if not
                                                             -- specified, case-sensitive.
                                            default-no  (3)} -- case-sensitivity is as specified on request, and if not
                                                             -- specified, not case-sensitive.
                                                        OPTIONAL}

ProcessingInformation ::= SEQUENCE{
  commonInfo            [0]  IMPLICIT CommonInfo OPTIONAL,
      -- Key elements follow:
  databaseName          [1]  IMPLICIT DatabaseName,
  processingContext     [2]  IMPLICIT INTEGER {
                                access                 (0),     -- e.g. choosing databases
                                search                 (1),     -- e.g. "search strategies" or search forms
                                retrieval              (2),     -- e.g. recommended element combinations
                                record-presentation    (3),     -- display of retrieved records
                                record-handling        (4)      -- handling (e.g. saving) of retrieved records
                },
```

```
 name                         [3]  IMPLICIT InternationalString,
 oid                          [4]  IMPLICIT OBJECT IDENTIFIER,
    -- No non-key brief elements
    -- Non-brief elements follow:
 description                  [5]  IMPLICIT HumanString OPTIONAL,
                                        -- use element set name 'description' to retrieve all except instructions.
 instructions                 [6]  IMPLICIT EXTERNAL OPTIONAL -- mandatory in full record
                                   }


VariantSetInfo ::= SEQUENCE {
         -- A record in this category describes a variant set definition, i.e., classes, types, and values, for a specific
         -- variant set definition supported by the target. Support by the target of a particular variant set definition
         -- does not imply that the definition is supported for any specific database or element.
 commonInfo            [0] IMPLICIT CommonInfo OPTIONAL,
    -- Key elements follow:
 variantSet        [1] IMPLICIT OBJECT IDENTIFIER,
    -- Non-key brief elements follow:
 name              [2] IMPLICIT InternationalString,
    -- Non-brief elements follow:
 variants          [3] IMPLICIT SEQUENCE OF VariantClass OPTIONAL
                              -- mandatory in full record
                               }

    -- Subsidiary structures for VariantSetInfo
         VariantClass ::= SEQUENCE {
              name               [0] IMPLICIT InternationalString OPTIONAL,
              description        [1] IMPLICIT HumanString OPTIONAL,
              variantClass       [2] IMPLICIT INTEGER,
              variantTypes       [3] IMPLICIT SEQUENCE OF VariantType}
         VariantType ::= SEQUENCE {
              name               [0] IMPLICIT InternationalString OPTIONAL,
              description        [1] IMPLICIT HumanString OPTIONAL,
              variantType        [2] IMPLICIT INTEGER,
              variantValue       [3] IMPLICIT VariantValue OPTIONAL}
         VariantValue ::= SEQUENCE {
              dataType           [0] PrimitiveDataType,
              values             [1] ValueSet OPTIONAL }
         ValueSet ::= CHOICE {
              range          [0] IMPLICIT ValueRange,
              enumerated     [1] IMPLICIT SEQUENCE OF ValueDescription }
         ValueRange ::= SEQUENCE {
                   -- At last one the following must be supplied, both may be supplied.
              lower          [0] ValueDescription OPTIONAL,
              upper          [1] ValueDescription OPTIONAL }
         ValueDescription ::= CHOICE{
              integer                 INTEGER,
              string                  InternationalString,
              octets                  OCTET STRING,
              oid                     OBJECT IDENTIFIER,
              unit               [1]  IMPLICIT Unit,
```

```
            valueAndUnit     [2]  IMPLICIT IntUnit
                -- oid and unit can't be used in a ValueRange
                        }



UnitInfo ::= SEQUENCE {
  commonInfo     [0] IMPLICIT CommonInfo OPTIONAL,
     -- Key elements follow:
  unitSystem     [1] IMPLICIT InternationalString,
     -- No non-key brief elements
     -- Non-brief elements follow:
  description    [2] IMPLICIT HumanString OPTIONAL,
  units          [3] IMPLICIT SEQUENCE OF UnitType OPTIONAL
                        -- mandatory in full record
                     }

     -- Subsidiary structures for UnitInfo
        UnitType ::= SEQUENCE {
            name         [0] IMPLICIT InternationalString OPTIONAL,
            description  [1] IMPLICIT HumanString OPTIONAL,
            unitType     [2] StringOrNumeric,
            units        [3] IMPLICIT SEQUENCE OF Units}

        Units ::= SEQUENCE {
            name         [0] IMPLICIT InternationalString OPTIONAL,
            description  [1] IMPLICIT HumanString OPTIONAL,
            unit         [2] StringOrNumeric}

CategoryList ::= SEQUENCE {
  commonInfo      [0] IMPLICIT CommonInfo OPTIONAL,
     -- Only one record expected per Explain database. All elements appear in brief presentation.
  categories     [1] IMPLICIT SEQUENCE OF CategoryInfo }
        CategoryInfo ::= SEQUENCE {
            category            [1] IMPLICIT InternationalString,
            originalCategory    [2] IMPLICIT InternationalString OPTIONAL,
            description         [3] IMPLICIT HumanString OPTIONAL,
            asn1Module          [4] IMPLICIT InternationalString OPTIONAL}


-- - - - - - - - - - - - - - Subsidiary definitions

CommonInfo ::= SEQUENCE {
  dateAdded              [0] IMPLICIT GeneralizedTime OPTIONAL,
  dateChanged            [1] IMPLICIT GeneralizedTime OPTIONAL,
  expiry                 [2] IMPLICIT GeneralizedTime OPTIONAL,
  humanString-Language   [3] IMPLICIT LanguageCode OPTIONAL,
     -- following not to occur for brief:
  otherInfo                  OtherInformation OPTIONAL}
```

```
HumanString ::= SEQUENCE OF SEQUENCE {
                    language      [0] IMPLICIT LanguageCode OPTIONAL,
                    text          [1] IMPLICIT InternationalString}

IconObject ::= SEQUENCE OF SEQUENCE{
              -- Note that the "SEQUENCE OF" is to allow alternative representations of the same Icon; it is not
              -- intended to allow multiple icons.
     bodyType [1] CHOICE{
                         ianaType     [1] IMPLICIT InternationalString,
                         z3950type    [2] IMPLICIT InternationalString,
                         otherType    [3] IMPLICIT InternationalString},
     content  [2] IMPLICIT OCTET STRING}


LanguageCode ::= InternationalString  -- from ANSI/NISO Z39.53-1994

ContactInfo ::= SEQUENCE {
  name            [0] IMPLICIT InternationalString OPTIONAL,
  description     [1] IMPLICIT HumanString OPTIONAL,
  address         [2] IMPLICIT HumanString OPTIONAL,
  email           [3] IMPLICIT InternationalString OPTIONAL,
  phone           [4] IMPLICIT InternationalString OPTIONAL}

NetworkAddress ::= CHOICE {
  internetAddress              [0] IMPLICIT SEQUENCE {
                                    hostAddress      [0]  IMPLICIT InternationalString,
                                    port             [1]  IMPLICIT INTEGER},
  osiPresentationAddress       [1] IMPLICIT SEQUENCE {
                                    pSel             [0] IMPLICIT InternationalString,
                                    sSel             [1] IMPLICIT InternationalString OPTIONAL,
                                    tSel             [2] IMPLICIT InternationalString OPTIONAL,
                                    nSap             [3] IMPLICIT InternationalString},
  other                        [2] IMPLICIT SEQUENCE {
                                    type         [0] IMPLICIT InternationalString,
                                    address      [1] IMPLICIT InternationalString}}

AccessInfo ::= SEQUENCE {
          -- AccessInfo contains the fundamental information about what facilities are required to use this target
          -- or server. For example, if an origin can handle none of the record syntaxes a database can provide,
          -- it might choose not to access the database.
  queryTypesSupported    [0]      IMPLICIT SEQUENCE OF QueryTypeDetails OPTIONAL,
  diagnosticsSets        [1]      IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
  attributeSetIds        [2]      IMPLICIT SEQUENCE OF AttributeSetId OPTIONAL,
  schemas                [3]      IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
  recordSyntaxes         [4]      IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
  resourceChallenges     [5]      IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
  restrictedAccess       [6]      IMPLICIT AccessRestrictions OPTIONAL,
  costInfo               [8]      IMPLICIT Costs OPTIONAL,
  variantSets            [9]      IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL,
  elementSetNames        [10]     IMPLICIT SEQUENCE OF ElementSetName OPTIONAL,
  unitSystems            [11]     IMPLICIT SEQUENCE OF InternationalString}
```

```
-- begin auxiliary definitions for AccessInfo
-- Begin Query Details
QueryTypeDetails ::= CHOICE {
  private            [0]      IMPLICIT PrivateCapabilities,
  rpn                [1]      IMPLICIT RpnCapabilities,
  iso8777            [2]      IMPLICIT Iso8777Capabilities,
  z39-58             [100]    IMPLICIT HumanString,
  erpn               [101]    IMPLICIT RpnCapabilities,
  rankedList         [102]    IMPLICIT HumanString}

PrivateCapabilities ::= SEQUENCE {
  operators        [0]  IMPLICIT SEQUENCE OF SEQUENCE {
                          operator      [0] IMPLICIT InternationalString,
                          description   [1] IMPLICIT HumanString OPTIONAL } OPTIONAL,
  searchKeys       [1]  IMPLICIT SEQUENCE OF SearchKey OPTIONAL,  -- field names that can be searched
  description      [2]  IMPLICIT SEQUENCE OF HumanString OPTIONAL }

RpnCapabilities ::= SEQUENCE {
  operators                         [0] IMPLICIT SEQUENCE OF INTEGER OPTIONAL,
                                            -- Omitted means all operators are supported.
  resultSetAsOperandSupported       [1] IMPLICIT BOOLEAN,
  restrictionOperandSupported       [2] IMPLICIT BOOLEAN,
  proximity                         [3] IMPLICIT ProximitySupport OPTIONAL}

Iso8777Capabilities ::= SEQUENCE {
  searchKeys       [0]  IMPLICIT SEQUENCE OF SearchKey,  -- field names that may be searched
  restrictions     [1]  IMPLICIT HumanString OPTIONAL
                          -- Omitted means supported, not specifying units.
                              }

ProximitySupport ::= SEQUENCE {
  anySupport       [0]   IMPLICIT BOOLEAN,
                      -- 'false' means no proximity support, in which case unitsSupported not supplied.
  unitsSupported   [1]   IMPLICIT SEQUENCE OF CHOICE{
                      known    [1] IMPLICIT INTEGER, -- values from KnownProximityUnit
                      private  [2] IMPLICIT SEQUENCE{
                                  unit          [0] IMPLICIT INTEGER,
                                  description   [1] HumanString OPTIONAL}} OPTIONAL}

SearchKey ::= SEQUENCE {
  searchKey        [0]  IMPLICIT InternationalString,
  description      [1]  IMPLICIT HumanString OPTIONAL }
-- End Query details
AccessRestrictions ::= SEQUENCE OF SEQUENCE {
  accessType            [0] INTEGER {
                          any                    (0),
                          search                 (1),
                          present                (2),
                          specific-elements      (3),
                          extended-services      (4),
                          by-database            (5)},
```

```
  accessText            [1]  IMPLICIT HumanString OPTIONAL,
  accessChallenges      [2]  IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL}

Costs ::= SEQUENCE {
  connectCharge         [0] IMPLICIT Charge OPTIONAL,     -- Per-connection charge
  connectTime           [1] IMPLICIT Charge OPTIONAL,     -- Time-based charge
  displayCharge         [2] IMPLICIT Charge OPTIONAL,     -- Per-record charge
  searchCharge          [3] IMPLICIT Charge OPTIONAL,     -- Per-search charge
  subscriptCharge       [4]  IMPLICIT Charge OPTIONAL,    -- Subscription charges
  otherCharges          [5] IMPLICIT SEQUENCE OF SEQUENCE{   -- Other charges
                            forWhat       [1]  IMPLICIT HumanString,
                            charge        [2]  IMPLICIT Charge} OPTIONAL}
          Charge ::= SEQUENCE{
               cost          [1] IMPLICIT IntUnit,
               perWhat       [2] IMPLICIT Unit OPTIONAL,
                             -- e.g. "second," "minute," "line," "record"...
               text          [3] IMPLICIT HumanString OPTIONAL}
-- End Auxiliary definitions for AccessInfo

DatabaseList ::= SEQUENCE OF DatabaseName

AttributeCombinations ::= SEQUENCE {
  defaultAttributeSet  [0] IMPLICIT AttributeSetId,
                                 -- Default for the combinations. Also probably a good choice for the default
                                 -- in searches, but that isn't required.
  legalCombinations   [1] IMPLICIT SEQUENCE OF AttributeCombination }


AttributeCombination ::= SEQUENCE OF AttributeOccurrence
                            -- An AttributeCombination is a pattern for legal combination of attributes


AttributeOccurrence ::= SEQUENCE {
               -- An AttributeOccurrence lists the legal values for a specific attribute type in a combination.
      attributeSet          [0]  IMPLICIT AttributeSetId OPTIONAL,
      attributeType         [1]  IMPLICIT INTEGER,
      mustBeSupplied        [2]  IMPLICIT NULL OPTIONAL,
      attributeValues       CHOICE {
                                    any-or-none  [3] IMPLICIT NULL, -- All supported values are OK
                                    specific     [4] IMPLICIT SEQUENCE OF StringOrNumeric}}
                                                         -- Only these values allowed
END
```

RecordSyntax-SUTRS

```
{Z39-50-recordSyntax SUTRS (101)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString  FROM Z39-50-APDU-1995;
        SutrsRecord ::=  InternationalString
-- Line terminator is ASCII LF (X'0A').
-- Recommended maximum line length is 72 characters.
END
```

RecordSyntax-opac

```
{Z39-50-recordSyntax opac (102)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString  FROM Z39-50-APDU-1995;
OPACRecord ::= SEQUENCE {
          bibliographicRecord       [1]      IMPLICIT EXTERNAL OPTIONAL,
          holdingsData              [2]      IMPLICIT SEQUENCE OF HoldingsRecord OPTIONAL}
HoldingsRecord ::=  CHOICE {
          marcHoldingsRecord        [1]      IMPLICIT EXTERNAL,
          holdingsAndCirc           [2]      IMPLICIT HoldingsAndCircData}
HoldingsAndCircData ::=  SEQUENCE {
-- the following elements are required to display holdings in conformance with NISO standards.
          typeOfRecord              [1]      IMPLICIT InternationalString OPTIONAL, -- LDR 06
          encodingLevel             [2]      IMPLICIT InternationalString OPTIONAL, -- LDR 017
          format                    [3]      IMPLICIT InternationalString OPTIONAL, -- 007 00-01
          receiptAcqStatus          [4]      IMPLICIT InternationalString OPTIONAL, -- 008 06
          generalRetention          [5]      IMPLICIT InternationalString OPTIONAL, -- 008 12
          completeness              [6]      IMPLICIT InternationalString OPTIONAL, -- 008 16
          dateOfReport              [7]      IMPLICIT InternationalString OPTIONAL, -- 008 26-31
          nucCode                   [8]      IMPLICIT InternationalString OPTIONAL, -- 852 $a
          localLocation             [9]      IMPLICIT InternationalString OPTIONAL, -- 852 $b
          shelvingLocation          [10]     IMPLICIT InternationalString OPTIONAL, -- 852 $c
          callNumber                [11]     IMPLICIT InternationalString OPTIONAL, -- 852 $h and $i
          shelvingData              [12]     IMPLICIT InternationalString OPTIONAL, -- 852 $j thru $m
          copyNumber                [13]     IMPLICIT InternationalString OPTIONAL, -- 852 $t
          publicNote                [14]     IMPLICIT InternationalString OPTIONAL, -- 852 $z
          reproductionNote          [15]     IMPLICIT InternationalString OPTIONAL, -- 843
          termsUseRepro             [16]     IMPLICIT InternationalString OPTIONAL, -- 845
          enumAndChron              [17]     IMPLICIT InternationalString OPTIONAL, -- all 85x, 86x
          volumes                   [18]     IMPLICIT SEQUENCE OF Volume OPTIONAL,
                                                       -- repeats for each volume held
          circulationData           [19]     IMPLICIT SEQUENCE OF CircRecord OPTIONAL
                                                       -- repeats for each circulating item.
                                                       }
Volume ::=  SEQUENCE {
          enumeration               [1]      IMPLICIT InternationalString OPTIONAL,
          chronology                [2]      IMPLICIT InternationalString OPTIONAL,
          enumAndChron              [3]      IMPLICIT InternationalString OPTIONAL }
```

```
CircRecord ::=  SEQUENCE {
        availableNow            [1]     IMPLICIT BOOLEAN,
        availablityDate         [2]     IMPLICIT InternationalString OPTIONAL,
        availableThru           [3]     IMPLICIT InternationalString OPTIONAL,
        restrictions            [4]     IMPLICIT InternationalString OPTIONAL,
        itemId                  [5]     IMPLICIT InternationalString OPTIONAL,
        renewable               [6]     IMPLICIT BOOLEAN,
        onHold                  [7]     IMPLICIT BOOLEAN,
        enumAndChron            [8]     IMPLICIT InternationalString OPTIONAL,
        midspine                [9]     IMPLICIT InternationalString OPTIONAL,
        temporaryLocation       [10]    IMPLICIT InternationalString OPTIONAL}
END
```

RecordSyntax-summary

```
{Z39-50-recordSyntax summary (103)} DEFINITIONS ::=
BEGIN
IMPORTS OtherInformation, InternationalString  FROM Z39-50-APDU-1995;
BriefBib ::= SEQUENCE {
        title                   [1]     IMPLICIT InternationalString,
        author                  [2]     IMPLICIT InternationalString OPTIONAL,
        callNumber              [3]     IMPLICIT InternationalString OPTIONAL,
        recordType              [4]     IMPLICIT InternationalString OPTIONAL,
        bibliographicLevel      [5]     IMPLICIT InternationalString OPTIONAL,
        format                  [6]     IMPLICIT SEQUENCE OF FormatSpec OPTIONAL,
        publicationPlace        [7]     IMPLICIT InternationalString OPTIONAL,
        publicationDate         [8]     IMPLICIT InternationalString OPTIONAL,
        targetSystemKey         [9]     IMPLICIT InternationalString OPTIONAL,
        satisfyingElement       [10]    IMPLICIT InternationalString OPTIONAL,
        rank                    [11]    IMPLICIT INTEGER OPTIONAL,
        documentId              [12]    IMPLICIT InternationalString OPTIONAL,
        abstract                [13]    IMPLICIT InternationalString OPTIONAL,
        otherInfo                       OtherInformation OPTIONAL}

FormatSpec      ::=  SEQUENCE {
        type            [1]     IMPLICIT InternationalString,
        size            [2]     IMPLICIT INTEGER OPTIONAL,
        bestPosn        [3]     IMPLICIT INTEGER OPTIONAL}
END
```

```
RecordSyntax-generic  -- For detailed semantics, see Appendix RET.
{Z39-50-recordSyntax GRS-1 (105)} DEFINITIONS ::=
BEGIN
EXPORTS Variant;
IMPORTS IntUnit, Unit, InternationalString, StringOrNumeric, Term FROM Z39-50-APDU-1995;

GenericRecord ::= SEQUENCE OF TaggedElement
TaggedElement ::= SEQUENCE {
    tagType          [1] IMPLICIT INTEGER OPTIONAL,
                         -- If omitted, default should be supplied dynamically by tagSet-M;
                         -- otherwise it should be statically specified by the schema.
    tagValue         [2] StringOrNumeric,
    tagOccurrence    [3] IMPLICIT INTEGER OPTIONAL,
                         -- Occurrence within the database record, and relative to the parent. No
                         -- default; if omitted, target not telling or it is irrelevant.
    content          [4] ElementData,
    metaData         [5] IMPLICIT ElementMetaData OPTIONAL,
    appliedVariant   [6] IMPLICIT Variant OPTIONAL}

ElementData ::= CHOICE{
    octets           OCTET STRING,
    numeric          INTEGER,
    date             GeneralizedTime,
    ext              EXTERNAL,
    string           InternationalString,
    trueOrFalse      BOOLEAN,
    oid              OBJECT IDENTIFIER,
    intUnit                  [1]  IMPLICIT IntUnit,
    elementNotThere          [2] IMPLICIT NULL,  -- element requested but not there
    elementEmpty             [3] IMPLICIT NULL,  -- element there, but empty
    noDataRequested          [4] IMPLICIT NULL,  -- variant request said 'no data'
    diagnostic               [5] IMPLICIT EXTERNAL,
    subtree                  [6] SEQUENCE OF TaggedElement -- recursive, for nested tags
                             }


ElementMetaData ::= SEQUENCE{
    seriesOrder              [1]     IMPLICIT Order OPTIONAL, -- only for a non-leaf node
    usageRight               [2]     IMPLICIT Usage OPTIONAL,
    hits                     [3]     IMPLICIT SEQUENCE OF HitVector OPTIONAL,
    displayName              [4]     IMPLICIT InternationalString OPTIONAL,
                                         -- name for element that origin can use for display
    supportedVariants        [5]     IMPLICIT SEQUENCE OF Variant OPTIONAL,
    message                  [6]     IMPLICIT InternationalString OPTIONAL,
    elementDescriptor        [7]     IMPLICIT OCTET STRING OPTIONAL,
    surrogateFor             [8]     IMPLICIT TagPath OPTIONAL,
                                         -- the retrieved element is a surrogate for the element given by this path
    surrogateElement         [9]     IMPLICIT TagPath OPTIONAL,
                                         -- the element given by this path is a surrogate for the retrieved element
    other                    [99]    IMPLICIT EXTERNAL OPTIONAL}
```

```
        TagPath ::= SEQUENCE  OF SEQUENCE{
            tagType             [1]  IMPLICIT INTEGER OPTIONAL,
            tagValue            [2]  StringOrNumeric,
            tagOccurrence       [3]  IMPLICIT INTEGER OPTIONAL}



Order ::= SEQUENCE{
    ascending       [1] IMPLICIT BOOLEAN,
                            -- "true" means monotonically increasing (i.e. non-decreasing);
                            -- "false" means monotonically decreasing (i.e. non-increasing).
    order           [2] IMPLICIT INTEGER
                            -- Same as defined by 'elementOrdering' in tagSet-M, though this may be
                            -- overridden by schema.
                    }



Usage ::= SEQUENCE    {
    type            [1] IMPLICIT INTEGER{
                            redistributable     (1),    -- Element is freely redistributable.
                            restricted          (2),    -- Restriction contains statement.
                            licensePointer      (3)     -- Restriction contains license pointer.
                            },
    restriction     [2] IMPLICIT InternationalString OPTIONAL}



HitVector ::= SEQUENCE{
        -- Each hit vector points to a fragment within the element, via location and/or token.
    satisfier                       Term OPTIONAL, -- sourceword, etc.
    offsetIntoElement       [1] IMPLICIT IntUnit OPTIONAL,
    length                  [2] IMPLICIT IntUnit OPTIONAL,
    hitRank                 [3] IMPLICIT INTEGER OPTIONAL,
    targetToken             [4] IMPLICIT OCTET STRING OPTIONAL
                                    -- Origin may use token subsequently within a variantRequest (in
                                    --  an elementRequest) to retrieve (or to refer to) the fragment.
                            }



Variant ::= SEQUENCE{
    globalVariantSetId      [1] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
                                    -- Applies to the triples below, when variantSetId omitted. If
                                    -- globalVariantSetId omitted, default applies. Default may be provided by
                                    -- the tagSet-M element defaultVariantSetId.
```

```
     triples                 [2]  IMPLICIT SEQUENCE OF SEQUENCE{
                                    variantSetId  [0]  IMPLICIT OBJECT IDENTIFIER OPTIONAL,
                                                             -- If omitted, globalVariantSetId (above)
                                                             -- applies, unless that too is omitted, in
                                                             -- which case, default used.
                                    class              [1]  IMPLICIT INTEGER,
                                    type               [2]  IMPLICIT INTEGER,
                                    value              [3]  CHOICE{
                                                             INTEGER,
                                                             InternationalString,
                                                             OCTET STRING,
                                                             OBJECT IDENTIFIER,
                                                             BOOLEAN,
                                                             NULL,
                                                        -- Following need context tags:
                                                          unit              [1] IMPLICIT Unit,
                                                          valueAndUnit      [2] IMPLICIT IntUnit}}}
END
```

```
RecordSyntax-ESTaskPackage
{Z39-50-recordSyntax ESTaskPackage (106)} DEFINITIONS ::=
BEGIN
IMPORTS Permissions, InternationalString, IntUnit, DiagRec FROM Z39-50-APDU-1995;

TaskPackage ::= SEQUENCE{
     packageType             [1]     IMPLICIT OBJECT IDENTIFIER,
                                           -- oid of specific ES definition
     packageName             [2]     IMPLICIT InternationalString OPTIONAL,
     userId                  [3]     IMPLICIT InternationalString OPTIONAL,
     retentionTime           [4]     IMPLICIT IntUnit OPTIONAL,
     permissions             [5]     IMPLICIT Permissions OPTIONAL,
     description             [6]     IMPLICIT InternationalString OPTIONAL,
     targetReference         [7]     IMPLICIT OCTET STRING OPTIONAL,
     creationDateTime        [8]     IMPLICIT GeneralizedTime OPTIONAL,
     taskStatus              [9]     IMPLICIT INTEGER{
                                           pending      (0),
                                           active       (1),
                                           complete     (2),
                                           aborted      (3)},
     packageDiagnostics      [10]    IMPLICIT SEQUENCE OF DiagRec OPTIONAL,
     taskSpecificParameters  [11]    IMPLICIT EXTERNAL
                                           -- Use oid for specific ES definition
                                           -- (same oid as packageType above)
                                           -- and select [2] "taskPackage."
          }
END
```

```
ResourceReport-Format-Resource-1
{Z39-50-resourceReport resource-1 (1)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString FROM Z39-50-APDU-1995;
--
ResourceReport ::= SEQUENCE{
estimates       [1]    IMPLICIT SEQUENCE OF Estimate,
message         [2]    IMPLICIT InternationalString}
--
Estimate ::= SEQUENCE{
type            [1]    IMPLICIT EstimateType,
value           [2]    IMPLICIT INTEGER,  -- the actual estimate
currency-code   [3]    IMPLICIT INTEGER OPTIONAL
                          -- code for representation of currencies defined in ISO 4217-1990.
                          -- Applicable only to monetary estimates.
                                            }
EstimateType ::= INTEGER{
currentSearchRecords       (1),   -- estimated no. records in current (incomplete) result set for search
finalSearchRecords         (2),   -- estimated no. records that will be in result set if search completes
currentPresentRecords      (3),   -- estimated number of records in current (incomplete) set of
                                  -- records to be returned on Present
finalPresentRecords        (4),   -- estimated number of records that will be in the set of records
                                  -- to be returned by Present if Present completes
currentOpTimeProcessing    (5),   -- processing time (in .001 CPU seconds) used by operation so far
finalOpTimeProcessing      (6),   -- estimated total processing time (in .001 CPU seconds) that will
                                  -- be used by this operation if it completes
currentAssocTime           (7),   -- estimated processing time used by association (in .001 CPU sec.)
currentOperationCost       (8),   -- estimated cost for this operation so far
finalOperationCost         (9),   -- estimated cost for this operation if it completes
currentAssocCost           (10),  -- estimated cost for this association so far
finalOpTimeElapsed         (11),  -- estimated elapsed time for operation if it completes (in .001 sec.)
percentComplete            (12), -- estimated percent complete
currentSearchAssocCost     (13),  -- estimated search cost for association so far
currentPresentAssocCost    (14),  -- estimated present cost for this association so far
currentConnectAssocCost    (15),  -- estimated connect time cost for association so far
currentOtherAssocCost      (16)   -- estimated other cost (not included in 13-15) for association so far
                           }
END
```

```
ResourceReport-Format-Resource-2
{Z39-50-resourceReport resource-2 (2)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString, StringOrNumeric, IntUnit FROM Z39-50-APDU-1995;
--
ResourceReport ::= SEQUENCE{
estimates     [1]   IMPLICIT SEQUENCE OF Estimate OPTIONAL,
message       [2]   IMPLICIT InternationalString OPTIONAL}
--
Estimate ::= SEQUENCE{
type       [1]   StringOrNumeric,
                   -- Numeric values of 1-16 are the same as used in Resource-1.
value      [2]   IMPLICIT IntUnit
                   -- When expressing currency:
                   --     unitSystem (of Unit) is 'z3950'    (case insensitive)
                   --     unitType is 'iso4217-1990'         (case insensitive)
                   --     unit is currency code from ISO 4217-1990.
}
END
```

AccessControlFormat-prompt-1

```
{Z39-50-accessControl prompt-1 (1)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString, DiagRec FROM Z39-50-APDU-1995;
--
PromptObject ::= CHOICE{
   challenge          [1] IMPLICIT Challenge,
   response           [2] IMPLICIT Response}

  Challenge ::= SEQUENCE OF SEQUENCE {
      promptId           [1] PromptId,
                             -- Target supplies a number (for an enumerated prompt) or string (for a non
                             -- -enumerated prompt), for each prompt, and the origin returns it in response, for
                             -- this prompt, so target may correlate the prompt response with the prompt.
      defaultResponse    [2] IMPLICIT InternationalString OPTIONAL,
      promptInfo         [3] CHOICE{
                             character    [1] IMPLICIT InternationalString,
                             encrypted    [2] IMPLICIT Encryption} OPTIONAL,
                             -- Information corresponding to an enumerated prompt. For example if 'type', within
                             -- PromptId, is 'copyright', then promptInfo may contain a copyright statement.
      regExpr            [4] IMPLICIT InternationalString OPTIONAL,
                             -- A regular expression that promptResponse should match. See IEEE 1003.2
                             -- Volume 1, Section 2.8 "Regular Expression Notation." For example if promptId
                             -- is "Year of publication," regExpr might be "19[89][0-9]|20[0-9][0-9]".
      responseRequired   [5] IMPLICIT NULL OPTIONAL,
      allowedValues      [6] IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
                             -- e.g. promptId="Desired color"; allowed = 'red', 'blue','Green'.
```

```
     shouldSave          [7] IMPLICIT NULL OPTIONAL,
                                  -- Target recommends that origin save the data that it prompts from the
                                  -- user corresponding to this prompt, because it is likely to be requested again (so
                                  -- origin might not have to prompt the user next time).
     dataType            [8] IMPLICIT INTEGER{
                                     integer         (1),
                                     date            (2),
                                     float           (3),
                                     alphaNumeric    (4),
                                     url-urn         (5),
                                     boolean         (6)} OPTIONAL,
                                        -- Target telling origin type of data it wants. E.g., if "date" is specified,
                                        -- presumably the origin will try to prompt something "date-like" from the user.
     diagnostic          [9] IMPLICIT EXTERNAL OPTIONAL
                                        -- Intended for repeat requests when there is an error the origin
                                        -- should report to the user from previous attempt.
                         }


 Response ::= SEQUENCE OF SEQUENCE {
     promptId         [1] PromptId,
                               -- Corresponds to a prompt in the challenge, or may be unprompted, for
                               -- example "newPassword." If unprompted, should be "enumerated."
                               -- If this responds to a non-enumerated prompt, then nonEnumeratedPrompt
                               -- should contain the prompt string from the challenge.
     promptResponse   [2] CHOICE{
                                  string          [1] IMPLICIT InternationalString,
                                  accept          [2] IMPLICIT BOOLEAN,
                                  acknowledge     [3] IMPLICIT NULL,
                                  diagnostic      [4] DiagRec,
                                  encrypted       [5] IMPLICIT Encryption}}


PromptId ::= CHOICE{
  enummeratedPrompt       [1] IMPLICIT SEQUENCE{
                                     type                 [1] IMPLICIT INTEGER{
                                                                 groupId        (0),
                                                                 userId         (1),
                                                                 password       (2),
                                                                 newPassword    (3),
                                                                 copyright      (4),
                                     -- When type on Challenge is 'copyright', promptInfo has text of
                                     -- copyright message to be displayed verbatim to the user. If
                                     -- promptResponse indicates 'acceptance', this indicates the user has been
                                     -- shown, and accepted, the terms of the copyright. This is not intended
                                     -- to be legally binding, but provides a good-faith attempt on
                                     -- the part of the target to inform the user of the copyright.
                                                                 sessionId         (5)},
                                     suggestedString      [2] IMPLICIT InternationalString OPTIONAL},
  nonEnumeratedPrompt     [2] IMPLICIT InternationalString}
```

```
 Encryption ::= SEQUENCE{
    cryptType       [1] IMPLICIT OCTET STRING OPTIONAL,
    credential      [2] IMPLICIT OCTET STRING OPTIONAL,
                            --random number, SALT, or other factor
    data            [3] IMPLICIT OCTET STRING}

END
```

AccessControlFormat-des-1

```
{Z39-50-accessControlFormat  des-1 (2)} DEFINITIONS ::=
BEGIN
       DES-RN-Object ::= CHOICE {
           challenge       [1]        IMPLICIT DRNType,
           response        [2]        IMPLICIT DRNType}
       DRNType ::= SEQUENCE{
                   userId              [1]        IMPLICIT OCTET STRING OPTIONAL,
                   salt                [2]        IMPLICIT OCTET STRING OPTIONAL,
                   randomNumber        [3]        IMPLICIT OCTET STRING}
END
```

AccessControlFormat-krb-1

```
{Z39-50-accessControlFormat  krb-1 (3)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString FROM Z39-50-APDU-1995;

     KRBObject ::= CHOICE {
       challenge   [1]    IMPLICIT KRBRequest,
       response    [2]    IMPLICIT KRBResponse}
     KRBRequest ::= SEQUENCE{
           service           [1]   IMPLICIT InternationalString,
           instance          [2]   IMPLICIT InternationalString OPTIONAL,
           realm             [3]   IMPLICIT InternationalString OPTIONAL}
        -- target requests a ticket for the given service, instance, and realm
     KRBResponse ::= SEQUENCE{
           userid            [1]   IMPLICIT InternationalString OPTIONAL,
           ticket            [2]  IMPLICIT OCTET STRING}
        -- origin responds with a ticket for the requested service
END
```

ESFormat-PersistentResultSet

```
{Z39-50-extendedService PersistentResultSet (1)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString FROM Z39-50-APDU-1995;
PersistentResultSet ::= CHOICE{
    esRequest       [1] IMPLICIT SEQUENCE{
            toKeep      [1] IMPLICIT NULL,
            notToKeep   [2] OriginPartNotToKeep OPTIONAL},
    taskPackage     [2] IMPLICIT SEQUENCE{
            originPart  [1] IMPLICIT NULL,
            targetPart  [2] TargetPart OPTIONAL}}
OriginPartNotToKeep ::= SEQUENCE{
   originSuppliedResultSet  [1] IMPLICIT InternationalString OPTIONAL,
              -- name of transient result set, supplied on request, mandatory unless function is 'delete'
   replaceOrAppend      [2] IMPLICIT INTEGER{  -- only if function is "modify"
              replace      (1),
              append       (2)} OPTIONAL}
TargetPart ::= SEQUENCE{
   targetSuppliedResultSet  [1] IMPLICIT InternationalString OPTIONAL,
              -- Name of transient result set, supplied by target, representing the persistent result set to which
              -- package pertains. Meaningful only when package  is presented. (i.e. not on ES response).
   numberOfRecords [2] IMPLICIT INTEGER OPTIONAL}
END
```

ESFormat-PersistentQuery

```
{Z39-50-extendedService PersistentQuery (2)} DEFINITIONS ::=
BEGIN
IMPORTS Query, InternationalString, OtherInformation FROM Z39-50-APDU-1995;
PersistentQuery ::= CHOICE{
    esRequest           [1] IMPLICIT SEQUENCE{
            toKeep      [1] OriginPartToKeep OPTIONAL,
            notToKeep   [2] OriginPartNotToKeep},
    taskPackage         [2] IMPLICIT SEQUENCE{
            originPart  [1] OriginPartToKeep OPTIONAL,
            targetPart  [2] TargetPart}}
OriginPartToKeep ::= SEQUENCE{
   dbNames              [2] IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
   additionalSearchInfo  [3] OtherInformation OPTIONAL}
OriginPartNotToKeep ::= CHOICE{
   package      [1] IMPLICIT InternationalString,
   query        [2] Query}
TargetPart ::= Query
END
```

ESFormat-PeriodicQuerySchedule

```
{Z39-50-extendedService PeriodicQuerySchedule (3)} DEFINITIONS ::=
BEGIN
IMPORTS Query, InternationalString, IntUnit FROM Z39-50-APDU-1995
ExportSpecification, Destination FROM ESFormat-ExportSpecification;
```

```
PeriodicQuerySchedule ::= CHOICE{
   esRequest          [1] IMPLICIT SEQUENCE{
                              toKeep       [1] OriginPartToKeep,
                              notToKeep    [2] OriginPartNotToKeep},
   taskPackage        [2] IMPLICIT SEQUENCE{
                              originPart   [1] OriginPartToKeep,
                              targetPart   [2] TargetPart}}

OriginPartToKeep ::=SEQUENCE{
    activeFlag                  [1] IMPLICIT BOOLEAN,
    databaseNames               [2] IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
    resultSetDisposition        [3] IMPLICIT INTEGER{
                                        replace      (1),
                                        append       (2),
                                        createNew    (3)  -- Only if origin and target have agreement about
                                                          -- naming convention for the resulting package,
                                                          -- and only if no result set is specified.
                                } OPTIONAL,   -- Mandatory on 'create' if result set is specified, in
                                              -- which case it must be 'replace' or 'append.
    alertDestination            [4] Destination OPTIONAL,
    exportParameters            [5] CHOICE{
                                        packageName     [1]  IMPLICIT InternationalString,
                                        exportPackage   [2] ExportSpecification} OPTIONAL}

OriginPartNotToKeep ::= SEQUENCE{
    querySpec                   [1] CHOICE{
                                        actualQuery     [1] Query,
                                        packageName     [2] IMPLICIT InternationalString} OPTIONAL,
                                    -- mandatory for 'create'
    originSuggestedPeriod       [2] Period OPTIONAL,  -- mandatory for 'create'
    expiration                  [3] IMPLICIT GeneralizedTime OPTIONAL,
    resultSetPackage            [4] IMPLICIT InternationalString OPTIONAL}

TargetPart ::= SEQUENCE{
    actualQuery             [1] Query,
    targetStatedPeriod      [2] Period,
                                -- Target supplies the period, which may be same as origin proposed.
    expiration              [3] IMPLICIT GeneralizedTime OPTIONAL,
                                -- Target supplies value for task package. It may be the same as origin
                                -- proposed or different from (and overrides) origin proposal, but if
                                -- omitted, there is no expiration.
    resultSetPackage        [4] IMPLICIT InternationalString OPTIONAL,
                                -- May be omitted only if exportParameters was supplied. Target
                                -- supplies same name as origin supplied, if origin did supply a name.
    lastQueryTime           [5] IMPLICIT GeneralizedTime,
    lastResultNumber        [6] IMPLICIT INTEGER,
    numberSinceModify       [7] IMPLICIT INTEGER OPTIONAL}
```

```
Period ::= CHOICE{
        unit                [1] IMPLICIT IntUnit,
        businessDaily       [2] IMPLICIT NULL,
        continuous          [3] IMPLICIT NULL,
        other               [4] IMPLICIT InternationalString}
END
```

ESFormat-ItemOrder

```
{Z39-50-extendedService ItemOrder (4)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString FROM Z39-50-APDU-1995;
ItemOrder ::= CHOICE{
   esRequest     [1] IMPLICIT SEQUENCE{
                        toKeep        [1] OriginPartToKeep OPTIONAL,
                        notToKeep     [2] OriginPartNotToKeep},
   taskPackage   [2] IMPLICIT SEQUENCE{
                        originPart    [1] OriginPartToKeep OPTIONAL,
                        targetPart    [2] TargetPart}}
OriginPartToKeep ::= SEQUENCE{
    supplDescription     [1] IMPLICIT EXTERNAL OPTIONAL,
    contact              [2] IMPLICIT SEQUENCE{
                              name    [1]   IMPLICIT InternationalString OPTIONAL,
                              phone   [2]   IMPLICIT InternationalString OPTIONAL,
                              email   [3]   IMPLICIT InternationalString OPTIONAL} OPTIONAL,
    addlBilling          [3] IMPLICIT SEQUENCE{
                           paymentMethod       [1]   CHOICE{
                              billInvoice                    [0] IMPLICIT NULL,
                                 prepay                      [1] IMPLICIT NULL,
                                 depositAccount              [2] IMPLICIT NULL,
                                 creditCard                  [3] IMPLICIT CreditCardInfo,
                                 cardInfoPreviouslySupplied  [4] IMPLICIT NULL,
                                 privateKnown                [5] IMPLICIT NULL,
                                 privateNotKnown             [6] IMPLICIT EXTERNAL},
                           customerReference   [2] IMPLICIT InternationalString OPTIONAL,
                           customerPONumber    [3] IMPLICIT InternationalString OPTIONAL}
                                                        OPTIONAL}
CreditCardInfo    ::= SEQUENCE{
        nameOnCard        [1] IMPLICIT InternationalString,
        expirationDate    [2] IMPLICIT InternationalString,
        cardNumber        [3] IMPLICIT InternationalString}

OriginPartNotToKeep ::= SEQUENCE{ -- Corresponds to 'requestedItem' in service definition.
            --   Must supply at least one, and may supply both.
    resultSetItem          [1]   IMPLICIT SEQUENCE{
                              resultSetId        [1]   IMPLICIT InternationalString,
                              item               [2]   IMPLICIT INTEGER} OPTIONAL,
    itemRequest            [2] IMPLICIT EXTERNAL OPTIONAL
                              -- When itemRequest is an ILL-Request APDU,
                              -- use OID {iso standard 10161 abstract-syntax (2) ill-apdus (1)}
                        }
```

```
TargetPart ::= SEQUENCE{
    itemRequest           [1]  IMPLICIT EXTERNAL OPTIONAL,
        -- When itemRequest is an ILL-Request APDU, use OID 1.0.10161.2.1 (as above)
    statusOrErrorReport   [2] IMPLICIT EXTERNAL OPTIONAL,
        -- When statusOrErrorReport is an ILL Status-Or-Error-Report APDU, use OID  1.0.10161.2.1 (as above)
    auxiliaryStatus       [3]  IMPLICIT INTEGER{
                                notReceived         (1),
                                loanQueue           (2),
                                forwarded           (3),
                                unfilledCopyright   (4),
                                filledCopyright     (5)} OPTIONAL}
END
```

ESFormat-Update

```
{Z39-50-extendedService Update (5)} DEFINITIONS ::=
BEGIN
IMPORTS DiagRec, InternationalString FROM Z39-50-APDU-1995;
Update ::= CHOICE{
   esRequest      [1] IMPLICIT SEQUENCE{
                          toKeep        [1] OriginPartToKeep,
                          notToKeep     [2] OriginPartNotToKeep},
   taskPackage    [2] IMPLICIT SEQUENCE{
                          originPart    [1] OriginPartToKeep,
                          targetPart    [2] TargetPart}}


OriginPartToKeep ::= SEQUENCE{
    action                 [1] IMPLICIT INTEGER{
                                  recordInsert      (1),
                                  recordReplace     (2),
                                  recordDelete      (3),
                                  elementUpdate     (4)},
    databaseName           [2] IMPLICIT InternationalString,
    schema                 [3] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
    elementSetName         [4] IMPLICIT InternationalString OPTIONAL}


OriginPartNotToKeep ::= SuppliedRecords


TargetPart ::= SEQUENCE{
    updateStatus          [1] IMPLICIT INTEGER{
                                  success  (1),
                                  partial  (2),
                                  failure  (3)},
    globalDiagnostics     [2] IMPLICIT SEQUENCE OF DiagRec OPTIONAL,
                              -- These are non-surrogate diagnostics relating to the task,
                              -- not to individual records.
```

```
     taskPackageRecords   [3] IMPLICIT SEQUENCE OF TaskPackageRecordStructure
                                  -- There should be a TaskPackageRecordStructure for every record
                                  -- supplied. The target should create such a structure for every
                                  -- record immediately upon creating the task package to include
                                  -- correlation information and status. The record itself would not
                                  -- be included until processing for that record is complete.
                          }

-- Auxiliary definitions for Update
SuppliedRecords ::= SEQUENCE OF SEQUENCE{
     recordId          [1] CHOICE{
                              number  [1] IMPLICIT INTEGER,
                              string  [2] IMPLICIT InternationalString,
                              opaque  [3]  IMPLICIT OCTET STRING} OPTIONAL,
     supplementalId    [2] CHOICE{
                              timeStamp        [1] IMPLICIT GeneralizedTime,
                              versionNumber    [2] IMPLICIT InternationalString,
                              previousVersion  [3] IMPLICIT EXTERNAL} OPTIONAL,
     correlationInfo   [3] IMPLICIT CorrelationInfo OPTIONAL,
     record            [4] IMPLICIT EXTERNAL}

CorrelationInfo ::= SEQUENCE{
                       -- origin may supply one or both for any record:
     note     [1] IMPLICIT InternationalString OPTIONAL,
     id       [2] IMPLICIT INTEGER OPTIONAL}

TaskPackageRecordStructure ::= SEQUENCE{
          recordOrSurDiag      [1] CHOICE {
                                  record        [1] IMPLICIT EXTERNAL,
                                                    -- Choose 'record' if recordStatus is 'success', and
                                                    -- elementSetName was supplied.
                                  diagnostic    [2] DiagRec
                                                    -- Choose 'diagnostic', if RecordStatus is failure.
                                                              } OPTIONAL,
                                  -- The parameter recordOrSurDiag will thus be omitted only if
                                  -- 'elementSetName' was omitted and recordStatus is
                                  -- 'success'; or if record status is 'queued' or in 'process'.
          correlationInfo      [2] IMPLICIT CorrelationInfo OPTIONAL,
                                  -- This should be included if it was supplied by the origin.
          recordStatus         [3] IMPLICIT INTEGER{
                                       success      (1),
                                       queued       (2),
                                       inProcess    (3),
                                       failure      (4)}}
END
```

ESFormat-ExportSpecification

```
{Z39-50-extendedService ExportSpecification (6)} DEFINITIONS ::=
BEGIN
EXPORTS ExportSpecification, Destination; IMPORTS CompSpec, InternationalString FROM Z39-50-APDU-1995;
```

```
ExportSpecification ::= CHOICE{
   esRequest          [1] IMPLICIT SEQUENCE{
                  toKeep       [1] OriginPartToKeep,
                  notToKeep    [2] IMPLICIT NULL},
   taskPackage        [2] IMPLICIT SEQUENCE{
                  originPart   [1] OriginPartToKeep,
                  targetPart   [2] IMPLICIT NULL}}
OriginPartToKeep ::= SEQUENCE{
    composition                [1] IMPLICIT CompSpec,
    exportDestination          [2] Destination}

  Destination ::= CHOICE{
    phoneNumber         [1]     IMPLICIT InternationalString,
    faxNumber           [2]     IMPLICIT InternationalString,
    x400address         [3]     IMPLICIT InternationalString,
    emailAddress        [4]     IMPLICIT InternationalString,
    pagerNumber         [5]     IMPLICIT InternationalString,
    ftpAddress          [6]     IMPLICIT InternationalString,
    ftamAddress         [7]     IMPLICIT InternationalString,
    printerAddress      [8]     IMPLICIT InternationalString,
    other               [100]   IMPLICIT SEQUENCE{
                                    vehicle          [1]  IMPLICIT InternationalString  OPTIONAL,
                                    destination      [2] IMPLICIT InternationalString}}
END
```

ESFormat-ExportInvocation

```
{Z39-50-extendedService ExportInvocation (7)} DEFINITIONS ::=
BEGIN
IMPORTS InternationalString, IntUnit FROM Z39-50-APDU-1995
ExportSpecification FROM ESFormat-ExportSpecification;
ExportInvocation ::= CHOICE{
   esRequest          [1] IMPLICIT SEQUENCE{
                  toKeep       [1] OriginPartToKeep,
                  notToKeep    [2] OriginPartNotToKeep},
   taskPackage        [2] IMPLICIT SEQUENCE{
                  originPart   [1] OriginPartToKeep,
                  targetPart   [2] TargetPart OPTIONAL}}

OriginPartToKeep ::= SEQUENCE{
    exportSpec         [1] CHOICE{
                              packageName         [1] IMPLICIT InternationalString,
                              packageSpec         [2] ExportSpecification},
    numberOfCopies       [2] IMPLICIT INTEGER}
```

```
OriginPartNotToKeep      ::= SEQUENCE{
    resultSetId          [1] IMPLICIT InternationalString,
    records              [2] CHOICE{
                             all      [1] IMPLICIT NULL,
                             ranges   [2]   IMPLICIT SEQUENCE OF SEQUENCE{
                                            start     [1] IMPLICIT INTEGER,
                                            count     [2] IMPLICIT INTEGER OPTIONAL
                                                -- Count may be omitted only on last range, to indicate
                                                -- "all remaining records beginning with 'start'."
                                      }}}

TargetPart    ::= SEQUENCE{
        estimatedQuantity        [1] IMPLICIT IntUnit OPTIONAL,
        quantitySoFar            [2] IMPLICIT IntUnit OPTIONAL,
        estimatedCost            [3] IMPLICIT IntUnit OPTIONAL,
        costSoFar                [4] IMPLICIT IntUnit OPTIONAL}
END
```

UserInfoFormat-searchResult-1

```
{Z39-50-userInfoFormat searchResult-1 (1)} DEFINITIONS ::=
BEGIN
IMPORTS DatabaseName, Term, Query, IntUnit, InternationalString FROM Z39-50-APDU-1995;
SearchInfoReport ::= SEQUENCE OF SEQUENCE{
    subqueryId               [1] IMPLICIT InternationalString OPTIONAL,
                                   -- shorthand identifier of subquery
    fullQuery                [2] IMPLICIT BOOLEAN,             -- 'true' means this is the full query; 'false',
                                                               -- a sub-query
    subqueryExpression       [3] QueryExpression OPTIONAL,     -- A subquery of the query as
                                                               -- submitted. May be whole query;
                                                               -- if so, "fullQuery" should be 'true'.
    subqueryInterpretation   [4] QueryExpression OPTIONAL,     -- how target interpreted subquery
    subqueryRecommendation   [5] QueryExpression OPTIONAL,     -- target-recommended alternative
    subqueryCount            [6] IMPLICIT INTEGER OPTIONAL,        -- Number of records for this
                                                               -- subQuery, across all of the specified
                                                               -- databases. (If during search, via resource
                                                               -- control, number of records so far).
    subqueryWeight           [7] IMPLICIT IntUnit OPTIONAL,    -- relative weight of this subquery
    resultsByDB              [8] IMPLICIT ResultsByDB OPTIONAL}

ResultsByDB ::= SEQUENCE OF SEQUENCE{
    databases            [1] CHOICE{
                             all        [1] IMPLICIT NULL,
                                            -- applies across all of the databases in Search PDU
                             list       [2]   IMPLICIT SEQUENCE OF DatabaseName
                                            -- applies across all databases in this list
                         },
    count                [2] IMPLICIT INTEGER OPTIONAL,
                             -- Number of records for query component (and, as above, if during search,
                             -- via resource control, number of records so far).
```

```
    resultSetName        [3] IMPLICIT InternationalString OPTIONAL
                             -- Target-assigned result set by which subQuery is available. Should not
                             -- be provided unless processing for this query component is concluded (i.e.,
                             -- when this report comes during search, via resource control, as opposed
                             -- to after search, via additionalSearchInfo).
                                  }

QueryExpression ::=  CHOICE {
               term       [1] IMPLICIT SEQUENCE{
                                   queryTerm       [1]   Term,
                                   termComment     [2] IMPLICIT InternationalString OPTIONAL},
               query      [2] Query}
END



{Z39-50-elementSpec eSpec-1 (1)} DEFINITIONS ::=
BEGIN
IMPORTS Variant FROM RecordSyntax-generic
StringOrNumeric, InternationalString FROM Z39-50-APDU-1995;
--
Espec-1 ::= SEQUENCE{
    elementSetNames          [1] IMPLICIT SEQUENCE OF InternationalString OPTIONAL,
                                     -- Origin may include one or more element set names, each
                                     -- specifying a set of elements. Each of the elements is to be
                                     -- treated as an elementRequest in the form of simpleElement,
                                     -- where occurrence is 1.
    defaultVariantSetId      [2] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
                                     -- If supplied, applies whenever variantRequest
                                     -- does not include variantSetId.
    defaultVariantRequest    [3] IMPLICIT Variant OPTIONAL,
                                     -- If supplied, then for each simple elementRequest that does not
                                     -- include a variantRequest, the defaultVariantRequest applies.
                                     -- (defaultVariantRequest does not apply to a compositeRequest.)
    defaultTagType           [4] IMPLICIT INTEGER OPTIONAL,
                                     -- If supplied, applies whenever 'tagType' (within 'tag' within TagPath)
                                     -- is omitted.
    elements                 [5] IMPLICIT SEQUENCE OF ElementRequest OPTIONAL}
--

ElementRequest::= CHOICE{
    simpleElement        [1]   IMPLICIT SimpleElement,
    compositeElement     [2] IMPLICIT SEQUENCE{
                         elementList       [1] CHOICE{
                                     primitives          [1]   IMPLICIT SEQUENCE OF InternationalString,
                                                                  -- Origin may specify one or more element
                                                                  -- set names, each identifying a set of elements,
                                                                  -- and the composite element is the union.
                                     specs               [2] IMPLICIT SEQUENCE OF SimpleElement},
```

```
                        deliveryTag        [2] IMPLICIT TagPath,
                                                 -- DeliveryTag tagPath for compositeElement may not
                                                 -- include wildThing or wildPath.
                        variantRequest    [3] IMPLICIT Variant OPTIONAL}}

SimpleElement ::= SEQUENCE{
            path                    [1] IMPLICIT TagPath,
            variantRequest    [2] IMPLICIT Variant OPTIONAL}


TagPath ::= SEQUENCE OF CHOICE{
    specificTag  [1] IMPLICIT SEQUENCE{
                        tagType       [1] IMPLICIT INTEGER OPTIONAL,
                                              -- If omitted, then 'defaultTagType' (above) applies, if supplied, and
                                              -- if not supplied, then default listed in schema applies.
                        tagValue      [2] StringOrNumeric,
                        occurrence    [3] Occurrences OPTIONAL
                                              -- default is "first occurrence"
                                      },
    wildThing    [2] Occurrences,
                        -- Get Nth "thing" at this level, regardless of tag, for each N specified by
                        -- "Occurrences" (which may be 'all' meaning match every element at this level).
                        -- E.g., if "Occurrences" is 3, get third element regardless of its tag or the tag of
                        -- the first two elements.
    wildPath      [3] IMPLICIT NULL
                        -- Match any tag, at this level or below, that is on a path for which next tag in this
                        -- TagPath sequence occurs. WildPath may not be last member of the TagPath
                        -- sequence.
                                              }
--

Occurrences ::= CHOICE{
    all        [1] IMPLICIT NULL,
    last       [2] IMPLICIT NULL,
    values    [3] IMPLICIT SEQUENCE{
                    start          [1] IMPLICIT INTEGER,
                                        -- if 'start' alone is included, then single occurrence is requested
                    howMany    [2] IMPLICIT INTEGER OPTIONAL
                                        -- For example, if 'start' is 5 and 'howMany' is 6, then request is for
                                        -- "occurrences 5 through 10."
                                        }}
END
```