CTFile Formats

June 2005

June 2005

Contents

Chapter 8: RDfiles

Chapter 9: XDfiles

Chapter 10: The Extended Connection Table (V3000)

Chapter 11: The Extended Reaction File

Appendix A: Stereo Notes

Index

Chapter 1: Introduction

Elsevier MDL supports a number of file formats for representation and communication of chemical information. This document describes the formats for Elsevier MDL's CTfiles (chemical table files):

- Chapters 2 and 3 describe the Connection Table (V2000) format.

- Chapters 4 through 9 describe the standard CTfile formats.

- Chapter 10 describes the V3000 extended molfile format. All extended molfiles can be easily identified by the "V3000" version stamp in the header portion of the file. You are most likely to find the extended molfile format in CTfiles written from MDL ISIS/Host or MDL ISIS/Desktop version 2.0 or higher.

- Chapter 11 describes the V3000 extended rxnfile format.

Change Log

The following are the changes in this document:

Change	Page(s)
June 2005	
Product and company rebranding changes	
Fixed example in 3D feature type -5 in "The Connection Table [CTAB] (V2000)"	25
Added "Automatic V3000 Output" section in "The Extended Connection Table (V3000)"	98
October 2003	
Added new chapter on XDfile format	51
Added XDfile in this Introduction	7
August 2002	
Deleted chapter on moving CTfiles on/off the Clipboard	
Minor corrections	6, 8, 9, 13, 32, 40
Added new information on enhanced stereochemistry features	56
Added new chapter on Extended Reaction File format	62
May, 2001	
Added section describing advantages of V3000 file format	
Added section on V3000 Collection Block	
Minor corrections	
December, 1999	
Updated entries in "Atom List"	
December, 1998	
Updated "Example of an SDfile"	
August, 1998	
Added STBOX field	

June, 1997	
Added Atom Attachment Order	
Added new ATTCHORD field	
October, 1996	
Minor corrections	
Enhanced description of connection table properties block	
Added Sgroup bracket style	

Extended Molfile Format (V3000)

The Extended Molfile format (V3000) molfile format offers a number of advantages over the V2000 format:

- Consolidates property information for chemical objects

- Provides better support for new chemical properties or objects

- Removes fixed field widths to support large structures

- Uses free format and tagging of information for easier parsing

- Provides better support for backward compatibility through BEGIN/END blocks

Current Elsevier MDL products support reading and writing of *both* molfile formats. These products will also preferentially write V2000 molfiles to maximize interoperability with third party applications. However, the fixed limits and distributed property information in the V2000 format make it harder to add new planned representational enhancements. The V3000 format is intended to be the primary means for communication of future enhancements to MDL chemical representation features.

Note: There are situations when the V3000 format will automatically be written instead of the V2000 format. See .

Your code for writing molfiles should be able to export structure information in the V3000 format if any structural features are present which cannot be defined in the V2000 format.

Standard CTfiles

The following figure illustrates the relationship between the various file formats described below:
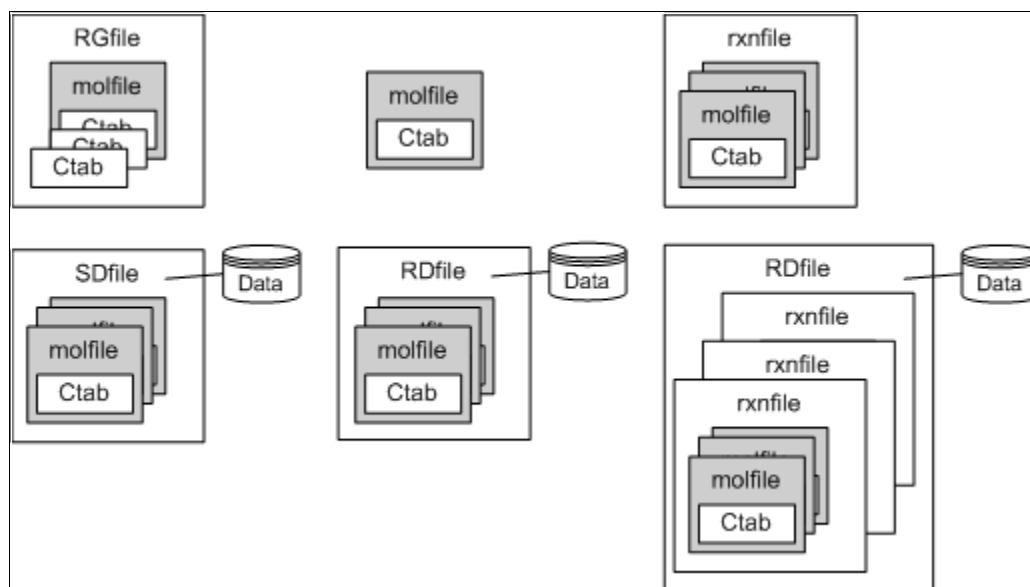
molfiles	Molecule files: Each molfile describes a single molecular structure which can contain disjoint fragments.
RGfiles	Rgroup files: An RGfile describes a single molecular query with Rgroups. Each RGfile is a combination of Ctabs defining the root molecule and each member of each Rgroup in the query.
rxnfiles	Reaction files: Each rxnfile contains the structural information for the reactants and products of a single reaction. Elsevier MDL currently supports only the **REACCS** type of rxnfile. The **CPSS** type of rxnfile written by CPSS programs is no longer supported and is not described in this document.
SDfiles	Structure-data files: An SDfile contains structures and data for any number of molecules. Together with RDfiles, SDfiles are the primary format for large-scale data transfer between MDL databases.
RDfiles	Reaction-data files: Similar to SDfiles in concept, the RDfile is a more general format that can include reactions as well as molecules, together with their associated data. Although RDfiles are used primarily by ISIS and REACCS, MACCS-II can also read and write RDfiles except for the reaction structure information (indicated by the square brackets in the MDL Program table).
XDfiles	XML-data files: XML-based data format for transferring recordsets of structure or reaction information with associated data. An XDfile can contain structures or reactions that use any of the CTfile formats, Chime strings, or SMILES strings. (Chime is an encrypted format that is used to render structures and reactions on a Web page. SMILES is a line notation format that uses character strings and SMILES, Simplified Molecule Input Line Entry System, syntax to represent a structure.)

The following figure illustrates three examples of XDfiles:

XDfile with molfile structures and associated data

XDfile with rxnfile reactions and associated data

XDfile with Chime strings and associated data
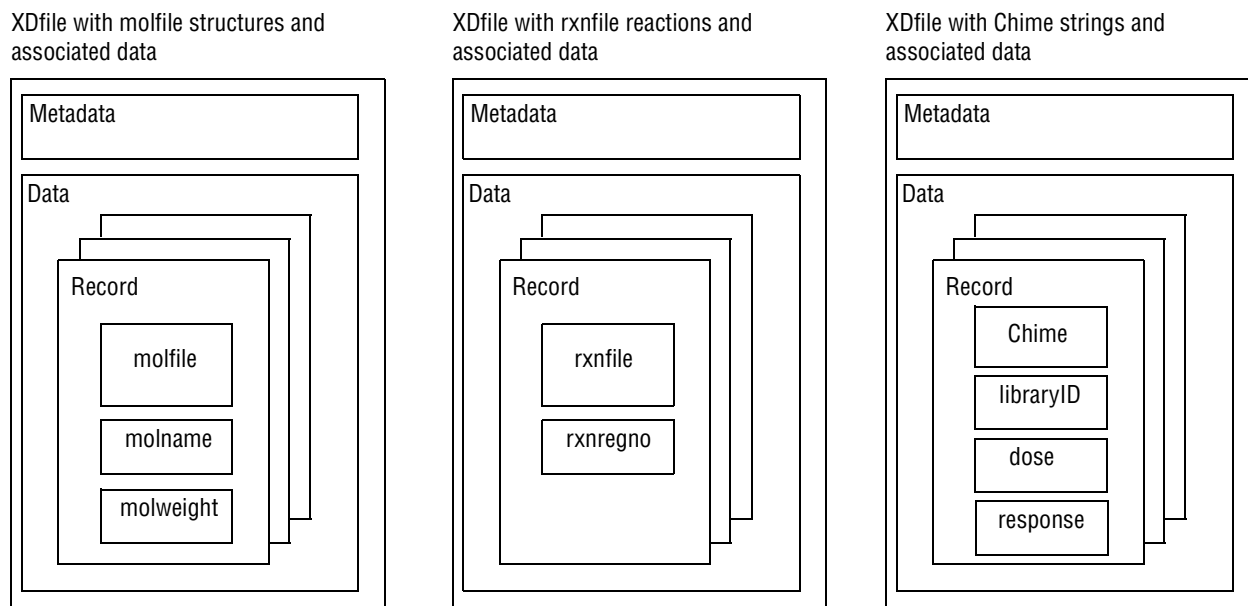
This table shows which CTfiles MDL programs can read and write.

Figure 1   MDL Programs

CTfile Type	MACCS-II	REACCS	ISIS	Core Interface
molfiles	+	+	+	+
RGfiles	+		+	+
rxnfiles		+	+	+
SDfiles	+		+	+
RDfiles	[+]	+	+	+
XDfiles				+

Some of the structural and query properties described in this document are generic in their applicability, while others are peculiar to certain CTfile types. The applicability of each property is identified in subsequent chapters by the bracketed terms shown in the following table.

Figure 2   Properties applicable to various CTfile types

Property	molfile	RGfile	SDfile	rxnfile	RDfile	XDfile
[Generic]	+	+	+	+	+	+ (mol/rxn)
[Sgroup]	+	+	+	[+]		+ (mol/[rxn])
[Rgroup]	+	+	+			+ (mol)
[3D]	+	+	+			+ (mol)
[Reaction]				+	+	+ (rxn)

Figure 2   Properties applicable to various CTfile types

[Query]	+	+		+		+ (mol/rxn)

Note: The XDfile inherits the functionality of the format of the embedded structure or reaction. In addition to the molfile and rxnfile formats, the XDfile supports Chime and SMILES strings.

Chapter 2: The Connection Table [CTAB] (V2000)

Overview

A connection table (Ctab) contains information describing the structural relationships and properties of a collection of atoms. The atoms may be wholly or partially connected by bonds. Such collections may, for example, describe molecules, molecular fragments, substructures, substituent groups, polymers, alloys, formulations, mixtures, and unconnected atoms. The connection table is fundamental to all of Elsevier MDL's file formats.

The following figure shows the connection table of a simple molecule (alanine) with the various data blocks identified. The connection table corresponds to the following alanine molecule. The atom numbers on the structure correspond to atom numbers in the Ctab. An atom number is assigned according to the order of the atom in the Atom Block.

Figure 3   Connection table organization illustrated using alanine


```
  6  5  0  0  1  0                  3 V2000
   -0.6622    0.5342    0.0000 C   0  0  2  0  0  0
    0.6622   -0.3000    0.0000 C   0  0  0  0  0  0
   -0.7207    2.0817    0.0000 C   1  0  0  0  0  0
   -1.8622   -0.3695    0.0000 N   0  3  0  0  0  0
    0.6220   -1.8037    0.0000 O   0  0  0  0  0  0
    1.9464    0.4244    0.0000 O   0  5  0  0  0  0
  1  2  1  0  0  0
  1  3  1  1  0  0
  1  4  1  0  0  0
  2  5  2  0  0  0
  2  6  1  0  0  0
M  CHG  2   4   1   6  -1
M  ISO  1   3  13
M  END
```

Blocks not used in this
Ctab: Atom List, Stext
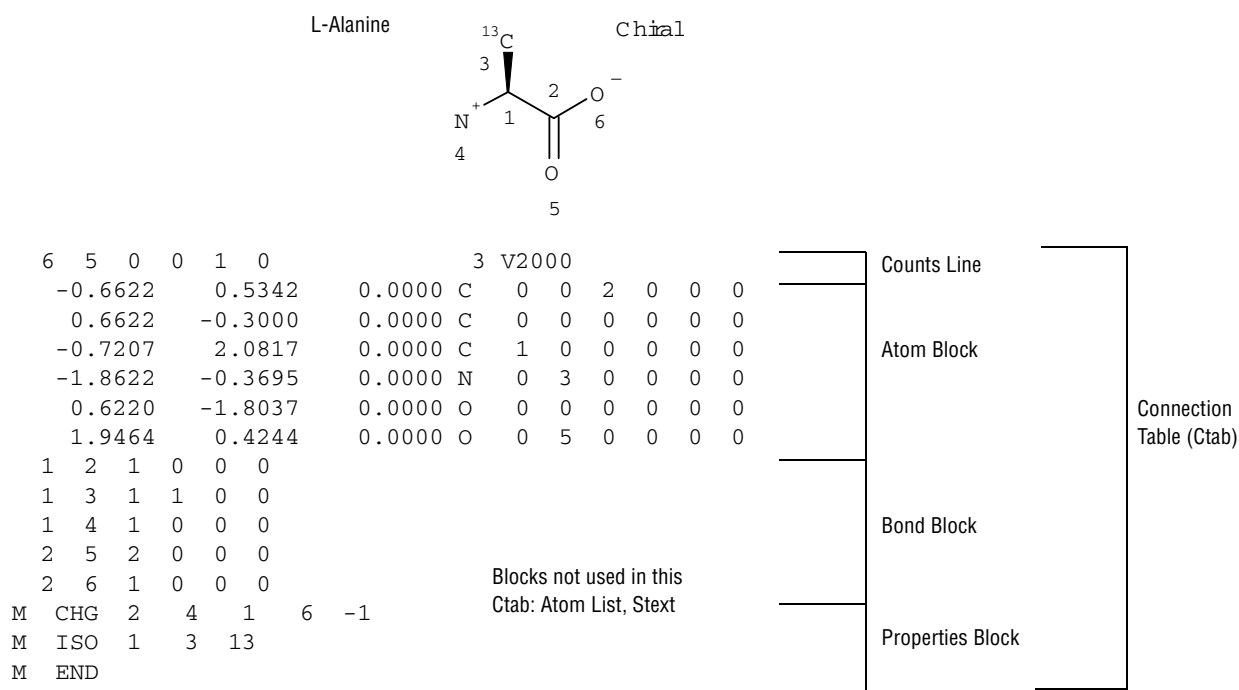
The format for a Ctab block is:

Counts line:	Important specifications here relate to the number of atoms, bonds, and atom lists, the chiral flag setting, and the Ctab version.
Atom block:	Specifies the atomic symbol and any mass difference, charge, stereochemistry, and associated hydrogens for each atom.
Bond block:	Specifies the two atoms connected by the bond, the bond type, and any bond stereochemistry and topology (chain or ring properties) for each bond.
Atom list block:	Identifies the atom (number) of the list and the atoms in the list.
Stext (structural text descriptor) block:	Used by MDL ISIS/Desktop programs.
Properties block:	Provides for future expandability of Ctab features, while maintaining compatibility with earlier Ctab configurations.

The detailed format for each block outlined above follows:

Note: A blank *numerical* entry on any line should be read as "0" (zero). Spaces are significant and correspond to one or more of the following:

- Absence of an entry

- Empty character positions within an entry

- Spaces between entries; single unless specifically noted otherwise

The FORTRAN format for coordinate information in the V2000 CTfile format is typically F10.4.

The Counts Line

```
aaabbblllfffcccsssxxxrrrpppiiimmmvvvvvv
```

Where:

aaa	= number of atoms (current max 255)*	**[Generic]**
bbb	= number of bonds (current max 255)*	**[Generic]**
lll	= number of atom lists (max 30)*	**[Query]**
fff	= (obsolete)	
ccc	= chiral flag: 0=not chiral, 1=chiral	**[Generic]**
sss	= number of stext entries	**[MDL ISIS/Desktop]**
xxx	= (obsolete)	
rrr	= (obsolete)	
ppp	= (obsolete)	
iii	= (obsolete)	
mmm	= number of lines of additional properties, including the M END line. No longer supported, the default is set to 999.	**[Generic]**

 * These limits apply to MACCS-II, REACCS, and the MDL ISIS/Host Reaction
Gateway, but not to the MDL ISIS/Host Molecule Gateway or MDL ISIS/Desktop.

For example, the counts line in the Ctab shown in the previous figure shows six atoms, five bonds, the CHIRAL flag *on*, and three lines in the properties block:

```
  6  5  0  0  1  0              3 V2000
```

The Atom Block

The Atom Block is made up of atom lines, one line per atom with the following format:

```
xxxxx.xxxxyyyyy.yyyyzzzzz.zzzz aaaddcccssshhhbbbvvvHHHrrriiimmmnnneee
```

where the values are described in the following table:

Figure 4   Meaning of values in the atom block

Field	Meaning	Values	Notes
x y z	atom coordinates		**[Generic]**
aaa	atom symbol	entry in periodic table or L for atom list, A, Q, * for unspecified atom, and LP for lone pair, or R# for Rgroup label	**[Generic, Query, 3D, Rgroup]**
dd	mass difference	-3, -2, -1, 0, 1, 2, 3, 4 (0 if value beyond these limits)	**[Generic]** Difference from mass in periodic table. Wider range of values allowed by `M  ISO` line, below. Retained for compatibility with older Ctabs, `M  ISO` takes precedence.
ccc	charge	0 = uncharged or value other than these, 1 = +3, 2 = +2, 3 = +1, 4 = doublet radical, 5 = -1, 6 = -2, 7 = -3	**[Generic]** Wider range of values in `M  CHG` and `M  RAD` lines below. Retained for compatibility with older Ctabs, `M  CHG` and `M  RAD` lines take precedence.
sss	atom stereo parity	0 = not stereo, 1 = odd, 2 = even, 3 = either or unmarked stereo center	**[Generic]** Ignored when read.
hhh	hydrogen count + 1	1 = H0, 2 = H1, 3 = H2, 4 = H3, 5 = H4	**[Query]** H0 means no H atoms allowed unless explicitly drawn. Hn means atom must have n or more H's in excess of explicit H's.
bbb	stereo care box	0 = ignore stereo configuration of this double bond atom, 1 = stereo configuration of double bond atom must match	**[Query]** Double bond stereochemistry is considered during SSS only if both ends of the bond are marked with stereo care boxes.
vvv	valence	0 = no marking (default) (1 to 14) = (1 to 14) 15 = zero valence	**[Generic]** Shows number of bonds to this atom, including bonds to implied H's.
HHH	H0 designator	0 = not specified, 1 = no H atoms allowed	**[MDL ISIS/Desktop]** Redundant with hydrogen count information. May be unsupported in future releases of Elsevier MDL software.
rrr	Not used		
iii	Not used		
mmm	atom-atom mapping number	1 - number of atoms	**[Reaction]**
nnn	inversion/retention flag	0 = property not applied 1 = configuration is inverted, 2 = configuration is retained,	**[Reaction]**
eee	exact change flag	0 = property not applied, 1 = change on atom must be exactly as shown	[Reaction, Query]

With Ctab version V2000, the dd and ccc fields have been superseded by the M ISO, M CHG, and M RAD lines in the properties block, described below. For compatibility, all releases since MACCS-II 2.0, REACCS 8.1, and MDL ISIS 1.0:

- Write appropriate values in both places if the values are in the old range.

- Use the atom block fields if there are no M ISO, M CHG, or M RAD lines in the properties block.

Support for these atom block fields may be removed in future releases of Elsevier MDL software.

The Bond Block

The Bond Block is made up of bond lines, one line per bond, with the following format:

`111222tttsssxxxrrrccc`

where the values are described in the following table:

Figure 5   Meaning of values in the bond block

Field	Meaning	Values	Notes
111	first atom number	1 - number of atoms	**[Generic]**
222	second atom number	1 - number of atoms	**[Generic]**
ttt	bond type	1 = Single, 2 = Double, 3 = Triple, 4 = Aromatic, 5 = Single or Double, 6 = Single or Aromatic, 7 = Double or Aromatic, 8 = Any	**[Query]** Values 4 through 8 are for SSS queries only.
sss	bond stereo	Single bonds: 0 = not stereo, 1 = Up, 4 = Either, 6 = Down, Double bonds: 0 = Use x-, y-, z-coords from atom block to determine cis or trans, 3 = Cis or trans (either) double bond	**[Generic]** The wedge (pointed) end of the stereo bond is at the first atom (Field 111 above)
xxx	not used		
rrr	bond topology	0 = Either, 1 = Ring, 2 = Chain	**[Query]** SSS queries only.
ccc	reacting center status	0 = unmarked, 1 = a center, -1 = not a center, Additional: 2 = no change, 4 = bond made/broken, 8 = bond order changes 12 = 4+8 (both made/broken and changes); 5 = (4 + 1), 9 = (8 + 1), and 13 = (12 + 1) are also possible	**[Reaction, Query]**

The Atom List Block   [Query]

Note: Newer programs use the M ALS item in the properties block in place of the atom list block. The atom list block is retained for compatibility, but information in an M ALS item supersedes atom list block information.

Made up of atom list lines, one line per list, with the following format:

`aaa kSSSSn 111 222 333 444 555`

where:

Field	Meaning
aaa	= number of atom (L) where list is attached
k	= T = [NOT] list, = F = normal list
n	= number of entries in list; maximum is 5
111...555	= atomic number of each atom on the list
S	= space

The Stext Block   [MDL ISIS/Desktop]

The Stext Block is made up of two-line entries with the following format:

```
xxxxx.xxxxyyyyy.yyyy
TTTT...
```

where:

Field	Meaning
x y	= stext coordinate
T	= stext text

The Properties Block

The Properties Block is made up of mmm lines of additional properties, where mmm is the number in the counts line described above. If a version stamp is present, mmm is ignored and the file is read until an "M   END" line is encountered. Currently mmm is no longer supported and is set to 999 as the default.

Most lines in the properties block are identified by a prefix of the form M   XXX where two spaces separate the M and XXX. Exceptions are:

- A   aaa, V   aaa vvvvvv,and G   aaappp, which indicate the following ISIS/Desktop properties: atom alias, atom value, and group abbreviation (called residue in ISIS), respectively.

- S   SKPnnn which causes the next nnn lines to be ignored.

The prefix: M   END terminates the properties block.

Variables in the formats can change properties but keep the same letter designation. For example, on the Charge, Radical, or Isotope lines, the "uniformity" of the vvv designates a general property identifier. On Sgroup property lines, the sss uniformity is used as an Sgroup index identifier.

All lines that are not understood by the program are ignored.

The descriptions below use the following conventions for values in field widths of 3:

```
n15     number of entries on line; value = 1 to 15
nn8     number of entries on line; value = 1 to 8
nn6     number of entries on line; value = 1 to 6
nn4     number of entries on line; value = 1 to 4
nn2     number of entries on line; value = 1 or 2
nn1     number of entries on line; value = 1
aaa     atom number; value = (1 to number of atoms)
```

The format for the properties included in this block follows. The format shows one entry; ellipses (. . .) indicate additional entries.

Atom Alias   [MDL ISIS/Desktop]

```
A  aaa
x...
```

aaa:	Atom number
x:	Alias text

Atom Value   [MDL ISIS/Desktop]

```
V  aaa v...
```

aaa:	Atom number
v:	Value text

Group Abbreviation   [MDL ISIS/Desktop]

```
G  aaappp
x...
```

aaa:	Atom number
ppp:	Atom number
x:	Abbreviation label

Abbreviation is required for compatibility with previous versions of MDL ISIS/Desktop which allowed abbreviations with only one attachment. The attachment is denoted by two atom numbers, aaa and ppp. All of the atoms on the aaa side of the bond formed by aaa-ppp are abbreviated. The coordinates of the abbreviation are the coordinates of aaa. The text of the abbreviation is on the following line (x...). In current versions of ISIS, abbreviations can have any number of attachments and are written out using the Sgroup appendixes. However, any ISIS abbreviations that do have one attachment are also written out in the old style, again for compatibility with older ISIS versions, but this behavior might not be supported in future versions.

Charge [Generic]

```
M  CHGnn8 aaa vvv ...
```

vvv:	-15 to +15. Default of 0 = uncharged atom. When present, this property supersedes all charge and radical values in the atom block, forcing a 0 charge on all atoms not listed in an M CHG or M RAD line.

Radical [Generic]

```
M  RADnn8 aaa vvv ...
```

vvv:	Default of 0 = no radical, 1 = singlet (:), 2 = doublet (. or ^), 3 = triplet (^^). When present, this property supersedes all charge and radical values in the atom block, forcing a 0 (zero) charge and radical on all atoms not listed in an M CHG or M RAD line.

Isotope [Generic]

```
M   ISOnn8 aaa vvv ...
```

vvv: Absolute mass of the atom isotope as a positive integer. When present, this property supersedes all isotope values in the atom block. Default (no entry) means natural abundance. The difference between this absolute mass value and the natural abundance value specified in the `PTABLE.DAT` file must be within the range of -18 to +12.

Ring Bond Count   [Query]

```
M   RBCnn8 aaa vvv ...
```

vvv: Number of ring bonds allowed: default of 0 = off, -1 = no ring bonds (r0),-2 = as drawn (r*); 2 = (r2), 3 = (r3), 4 or more = (r4).

Substitution Count   [Query]

```
M   SUBnn8 aaa vvv ...
```

vvv: Number of substitutions allowed: default of 0 = off, -1 = no substitution (s0),-2 = as drawn (s*); 1, 2, 3, 4, 5 = (s1) through (s5), 6 or more = (s6).

Unsaturated Atom [Query]

```
M   UNSnn8 aaa vvv ...
```

vvv: At least one multiple bond: default of 0 = off, 1 = on.

Link Atom [Query]

```
M   LINnn4 aaa vvv bbb ccc
```

vvv,bbb,ccc: Link atom (aaa) and its substituents, other than bbb and ccc, may be repeated 1 to vvv times, (vvv > = 2). The bbb and ccc parameters can be 0 (zero) for link nodes on atoms with attachment point information.

Atom List [Query]

```
M   ALS aaannn e 11112222333344445555...
```

aaa: Atom number, value

nnn: Number of entries on line (16 maximum)

e: Exclusion, value is T if a 'NOT' list, F if a normal list.

1111:   Atom symbol of list entry in field of width 4

Note: This line contains the atom symbol rather than the atom number used in the atom list block. Any data found in this item supersedes data from the atom list block. The number of entries can exceed the fixed limit of *5* in the atom list block entry.

Attachment Point [Rgroup]

```
M   APOnn2 aaa vvv ...
```

vvv: Indicates whether atom aaa of the Rgroup member is the first attachment point (vvv = 1), second attachment point (vvv = 2), both attachment points (vvv = 3); default of 0 = no attachment.

Atom Attachment Order   [Rgroup]

```
M   AAL aaann2 111 v1v 222 v2v ...
```

`aaa:`	Atom index of the Rgroup usage atom
`nn2:`	Number of pairs of entries that follow on the line
`111:`	Atom index of a neighbor of `aaa`
`v1v`	Attachment order for the `aaa-111` bond
`222`	Atom index of a neighbor of `aaa`
`v2v`	Attachment order for the `aaa-222` bond

Note: `v1v` and `v2v` are either 1 or 2 for the simple doubly attached Rgroup member.

This appendix provides explicit attachment list order information for R# atoms. The appendix contains atom neighbor index and atom neighbor value pairs. The atom neighbor value information identifies the atom neighbor index as the *nth* attachment. The implied ordering in V2000 molfiles is by atom index order for the neighbors of Rgroup usage atoms. If atom index order conflicts with the desired neighbor ordering at the R# atom, this appendix allows you to override to this default order.

If `v1v=1` and `v2v=2`, MDL ISIS/Host only writes this appendix if `111` is greater than `222`. Note, however, that the attachment values can be written in any order.

Rgroup Label Location   [Rgroup]

```
M   RGPnn8 aaa rrr ...
```

`rrr:`   Rgroup number, value from 1 to 32 *****, labels position of Rgroup on root.

* MACCS-II and ISIS/Desktop limit

Rgroup Logic, Unsatisfied Sites, Range of Occurrence   [Rgroup]

```
M   LOGnn1 rrr iii hhh ooo...
```

`rrr:`   Rgroup number, value from 1 to 32 *****

`iii:`   Number of another Rgroup which must only be satisfied if `rrr` is satisfied (IF `rrr` THEN `iii`)

`hhh:`   `RestH` property of Rgroup `rrr`; default is 0 = off, 1 = on. If this property is applied (on), sites labeled with Rgroup `rrr` may only be substituted with a member of the Rgroup or with H

`ooo`   Range of Rgroup occurrence required: `n` = exactly `n`, `n - m` = `n` through `m`, `> n` = greater than `n`, `< n` = fewer than `n`, default (blank) is `> 0`. Any non-contradictory combination of the preceding values is also allowed; for example:
`1, 3-7, 9, >11`.

* MACCS-II and ISIS/Desktop limit

Sgroup Type [Sgroup]

`M  STYnn8 sss ttt ...`

sss: Sgroup number

ttt: Sgroup type: SUP = superatom, MUL = multiple group, SRU = SRU
            type, MON = monomer, MER = Mer type, COP = copolymer, CRO =
            crosslink, MOD = modification, GRA = graft, COM = component, MIX
            = mixture, FOR = formulation, DAT = data Sgroup, ANY = any
            polymer, GEN = generic.

Note: For a given Sgroup, an STY line giving its type must appear before any other line that supplies information about it. For a data Sgroup, an SDT line must describe the data field before the SCD and SED lines that contain the data (see Data Sgroup Data below). When a data Sgroup is linked to another Sgroup, the Sgroup must already have been defined.

Sgroups can be in any order on the Sgroup Type line. Brackets are drawn around Sgroups with the M SDI lines defining the coordinates.
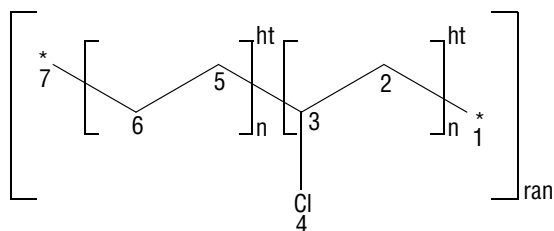
Sgroup Subtype   [Sgroup]

```
M   SSTnn8 sss ttt ...
```

ttt:	Polymer Sgroup subtypes: ALT = alternating, RAN = random, BLO = block

Figure 6   Ctab organization of an Sgroup structure

Polymer


```
GSMACCS-II10179110412D 1   0.00374    0.00000    0                    Header block (see Chapter 4 )

  7  6  0  0  0  0              16 V2000                               Counts line
    2.9463    0.3489    0.0000 *   0  0  0  0  0  0
    1.6126    1.1189    0.0000 C   0  0  0  0  0  0
    0.2789    0.3489    0.0000 C   0  0  3  0  0  0                     Atom block
    0.2789   -1.1911    0.0000 Cl  0  0  0  0  0  0
   -1.0548    1.1190    0.0000 C   0  0  0  0  0  0
   -2.3885    0.3490    0.0000 C   0  0  0  0  0  0
   -3.9246    1.1470    0.0000 *   0  0  0  0  0  0
  1  2  1  0  0  0
  2  3  1  0  0  0
  3  4  1  0  0  0           Number of entries on line
  5  6  1  0  0  0
  5  3  1  0  0  0                                                      Bond block
  7  6  1  0  0  0
M  STY  3   1 SRU   2 SRU   3 COP     ◄──── Type                        Atom list block
M  SST  1   3 RAN                     ◄──── Subtype                     Stext block
M  SLB  3   1   5   2   6   3   7     ◄──── Label
M  SCN  2   1 HT    2 HT              ◄──── Connectivity
M  SAL  1   2   5   6
M  SBL  1   2   5   6
M  SDI  1   4  -0.6103    1.2969   -0.6103    0.1710
M  SDI  1   4  -3.1565    0.1850   -3.1565    1.3110
M  SAL  2   3   2   3   4
M  SBL  2   2   1   5
M  SDI  2   4   2.2794    1.2969    2.2794    0.1709
M  SDI  2   4  -0.1657    0.1710   -0.1657    1.2969
M  SAL  3   7   1   2   3   4   5   6   7
M  SDI  3   4   3.6382    1.6391    3.6382   -1.7685
M  SDI  3   4  -4.7070   -1.7685   -4.7070    1.6391
M  END
```

Header block (see Chapter 4)

Counts line

Atom block

Bond block

Atom list block
Stext block

General Sgroup Info

Sgroup 1

Sgroup 2

Sgroup 3

Sgroup properties

Ctab block

Sgroup Labels   [Sgroup]

```
M   SLBnn8 sss vvv ...
```

vvv: Unique Sgroup identifier (for MACCS-II only, the integer label is from 1-512)

Sgroup Connectivity   [Sgroup]

```
M   SCNnn8 sss ttt ...
```

ttt:	HH = head-to-head, HT = head-to-tail, EU = either unknown. Left justified.

Sgroup Expansion   [Sgroup]

```
M   SDS EXPn15 sss ...
```

sss: Sgroup index of expanded superatoms

Sgroup Atom List   [Sgroup]

```
M   SAL sssn15 aaa ...
```

aaa: Atoms in Sgroup sss

Sgroup Bond List   [Sgroup]

```
M   SBL sssn15 bbb ...
```

bbb: Bonds in Sgroup sss. (For data Sgroups, bbb's are the containment bonds, for all other Sgroup types, bbb's are crossing bonds.)

Multiple Group Parent Atom List   [Sgroup]

```
M   SPA sssn15 aaa ...
```

aaa: Atoms in paradigmatic repeating unit of multiple group sss

Note: To ensure that all current molfile readers consistently interpret chemical structures, multiple groups are written in their fully expanded state to the molfile. The M   SPA atom list is a subset of the full atom list that is defined by the Sgroup Atom List M   SAL entry.

Sgroup Subscript   [Sgroup]

```
M   SMT sss m...
```

m...:   Text of subscript Sgroup sss. (For multiple groups, m... is the text representation of the multiple group multiplier. For superatoms, m... is the text of the superatom label.)

Sgroup Correspondence   [Sgroup]

```
M   CRS sssnn6 bb1 bb2 bb3
```

bb1, bb2:   Crossing bonds that share a common bracket

bb3:   Crossing bond in repeating unit that connect to bond bb1

Sgroup Display Information   [Sgroup]

```
M   SDI sssnn4 x1 y1 x2 y2
```

x1,y1,   Coordinates of bracket endpoints
x2,y2:

Superatom Bond and Vector Information   [Sgroup]

```
M  SBV sss bb1 x1 y1
```

bb1:   Bond connecting to contracted superatom

x1,y1:   Vector for bond bb1 connecting to contracted superatom sss

Data Sgroup Field Description   [Sgroup]

```
M  SDT sss fff...fffgghhh...hhhiijjj...
```

sss:	Index of data Sgroup
fff...fff:	30 character field name (in MACCS-II no blanks, commas, or hyphens)
gg:	Field type (in MACCS-II F = formatted, N = numeric, T = text)
hhh...hhh	20-character field units or format
ii:	Nonblank if data line is a query rather than Sgroup data, MQ = MACCS-II query, IQ = ISIS query, PQ = program name code query
jjj...:	Data query operator (blank for MACCS-II)

Data Sgroup Display Information   [Sgroup]

```
M  SDD sss xxxxx.xxxxyyyyy.yyyy eeefgh i jjjkkk ll m  noo
```

sss:	Index of data Sgroup
x,y:	Coordinates (2F10.4)
eee:	(Reserved for future use)
f:	Data display, A = attached, D = detached
g:	Absolute, relative placement, A = absolute, R = relative
h:	Display units, blank = no units displayed, U = display units
i:	(Reserved for future use)
jjj:	Number of characters to display (1...999 or ALL)
kkk:	Number of lines to display (unused, always 1)
ll:	(Reserved for future use)
m:	Tag character for tagged detached display (if non-blank)
n:	Data display DASP position (1...9). (MACCS-II only)
oo:	(Reserved for future use)

Data Sgroup Data   [Sgroup]

```
M  SCD sss d...
M  SED sss d...
```

d...:   Line of data for data Sgroup sss (69 chars per line, columns 12-80)

Note: A line of data is entered as text in 69-character substrings. Each SCD line adds 69 characters to a text buffer (starting with successive SCDs at character positions 1, 70, and 139). Following zero or more SCDs must be an SED, which may supply a final 69 characters. The SED initiates processing of the buffered line of text: trailing blanks are removed and right truncation to 200 characters is performed, numeric and formatted data are validated, and the line of data is added to data Sgroup sss. Left justification is not performed.

A data Sgroup may have more than one line of data, so more than one set of SCD and SED lines can be present for the same data Sgroup. The lines are added in the same order that they are encountered.

If 69 or fewer characters are to be entered on a line, they may be entered with a single SED not preceded by an SCD. On the other hand, if desired a line may be entered to a maximum of 3 SCDs followed by a blank SED that terminates the line. The set of SCD and SED lines describing one line of data for a given data Sgroup must appear together, with no intervening lines for other data Sgroups' data.

Sgroup Hierarchy Information   [Sgroup]

`M  SPLnn8 ccc ppp ...`

`ccc`:	Sgroup index of the child Sgroup)
`ppp`:	Sgroup index of the parent Sgroup (`ccc` and `ppp` must already be defined via an `STY` line prior to encountering this line)

Sgroup Component Numbers   [Sgroup]

`M  SNCnn8 sss ooo ...`

`sss`:	Index of component Sgroup
`ooo`:	Integer component order (1...256). This limit applies only to MACCS-II

3D Feature Properties   [3D]

`M  $3Dnnn`

`M  $3D...`:	See below for information on the properties block of a 3D molfile. These lines must all be contiguous

End of Block

`M  END`

This entry goes at the end of the properties block and is required for molfiles which contain a version stamp in the counts line.

The Properties Block for 3D Features   [3D]

For each 3D feature, the properties block includes:

- One 3D features count line
- One or more 3D features detail lines

The characters `M  $3D` appear at the beginning of each line describing a 3D feature. The information for 3D features starts in column 7.

Figure 7   Ctab organization of a 3D query


```
3D Query
  MACCS-II10179109553D 1   1.00000    0.00000     0

  8  7  0  0  0              18 V2000
    1.0252    0.2892    1.1122 C   0  0  0  0  0  0
   -0.4562    0.6578    1.3156 C   0  0  0  0  0  0
   -1.4813    0.3687    0.2033 C   0  0  0  0  0  0
   -1.0252   -0.2892   -1.1122 C   0  0  0  0  0  0
    0.4562   -0.6578   -1.3156 C   0  0  0  0  0  0
    1.4813   -0.3687   -0.2033 C   0  0  0  0  0  0
    4.1401   -0.1989    1.3456 N   0  0  0  0  0  0
    4.6453    0.5081    1.7417 C   0  0  0  0  0  0
  1  2  1  0  0  0
  2  3  2  0  0  0
  3  4  1  0  0  0
  4  5  2  0  0  0
  5  6  1  0  0  0
  6  1  2  0  0  0
  7  8  1  0  0  0
M  $3D  7
M  $3D -7  6
M  $3D  3
M  $3D  6  4  2
M  $3D -5 13
M  $3D  6    0.0000
M  $3D  1  2  3  4  5  6
M  $3D -8  7
M  $3D  9 10
M  $3D -3  6
M  $3D  9 11   -2.0000
M  $3D-16 12
M  $3D 12  1  0    1.5000
M  $3D-12 10
M  $3D 12  9  7   75.0000  105.0000
M  $3D -9  3
M  $3D  7  9    4.4000     5.7000
M  END
```

Header block (see Chapter 4)

Counts line

Atom block

Bond block

Atom list block

Stext block

Ctab block

3D Features Count

Centroid

Plane

Normal to Plane

Point

Exclusion sphere

Angle

Distance

Properties block

3D features count line

The first line in the properties block is the 3D features count line and has the following format:

M $3Dnnn

where nnn is the number of 3D features on a model.

3D features detail lines

The lines following the 3D features count line describe each 3D feature on a model. Each 3D feature description consists of an identification line and one or more data lines:

The identification line is the first line and contains the 3D feature's type identifier, color, and name.

Each data line describes the construction of the 3D feature.

Identification line

The 3D feature identification line has the following format:

```
M  $3Dfffccc aaa...aaa ttt...ttt
```

where the variables represent:

`fff:`	3D feature type
`ccc:`	Color number (an internal MDL number which is terminal dependent)
`aaa...aaa:`	3D feature name (up to 32 characters)
`ttt...ttt:`	Text comments (up to 32 characters) used by MDL programs (see 3D data constraints later in this chapter)

Figure 8   3D feature type identifiers

Identifier	Meaning
-1	Point defined by two points and a distance (in Angstroms)
-2	Point defined by two points and a percentage
-3	Point defined by a point, a normal line, and a distance
-4	Line defined by two or more points (a best fit line if more than two points)
-5	Plane defined by three or more points (a best fit plane if more than three points)
-6	Plane defined by a point and a line
-7	Centroid defined by points
-8	Normal line defined by a point and a plane
-9	Distance defined by two points and a range (in Angstroms)
-10	Distance defined by a point, line, and a range (in Angstroms)
-11	Distance defined by a point, plane, and a range (in Angstroms)
-12	Angle defined by three points and a range (in degrees)
-13	Angle defined by two intersecting lines and a range (in degrees)
-14	Angle defined by two intersecting planes and a range (in degrees)
-15	Dihedral angle defined by 4 points and a range (in degrees)
-16	Exclusion sphere defined by a point and a distance (in Angstroms)
-17	Fixed atoms in the model
nnn	A positive integer indicates atom or atom-pair data constraints

Data line

The 3D feature defines the data line format. Each 3D object is treated as a pseudoatom and identified in the connection table by a number. The 3D object numbers are assigned sequentially, starting with the next number greater than the number of atoms. The data line formats for the 3D feature types are:

Figure 9   3D feature type identifiers

Type	Description of Data Line

-1	The data line for a point defined by two points and a distance (Å) has the following format:
	
	`M   $3Diiijjjddddd.dddd`
	
	where the variables represent:
	
	`iii:` ID number of a point
	
	`jjj:` ID number of a second point
	
	`ddddd.dddd` Distance from first point in direction of second point (Å), 0 if not used
	
	The following example shows `POINT_1` created from the atoms 1 and 3 with a constraint distance of 2Å.
	
	The first line is the identification line. The second line is the data line.
	
	`M   $3D –1  4 POINT_1`
	`M   $3D  1  3    2.0000`

-2	The data line for a point defined by two points and a percentage has the format: `M   $3Diiijjjddddd.ddddd` where the variables represent: `iii` ID number of a point `jjj` ID number of a second point `ddddd.dddd` Distance (fractional) relative to distance between first and second points, 0 if not used
-3	The data line for a point defined by a point, a normal line, and a distance (Å) has the format: `M   $3Diiilllddddd.dddd` where the variables represent: `iii` ID number of a point `lll` ID number of a normal line `ddddd.dddd` Distance (Å), 0 if not used **Note:** For chiral models, the distance value is signed to specify the same or opposite direction of the normal.

-4	The data lines for a best fit line defined by two or more points have the following format:
| | ```
M  $3Dpppttttt.tttt
M  $3Diiijjj...zzz . . .
``` |
	where the variables represent:
	`ppp` Number of points defining the line
	`ttttt.tttt` Deviation (Å), 0 if not used.
	`iii` Each `iii`, `jjj`, and `zzz` is the ID number `jjj` of an item in the model that defines the line
	`jjj`
	`...`
	`zzz` (to maximum of 20 items per data line)
	The following line is defined by the four points 1, 14, 15, and 19 and has a deviation of 1.2Å. The first line is the identification line. The second and third lines are the data lines.
| | ```
M  $3D -4  2 N_TO_AROM
M  $3D  4    1.2000
M  $3D  1 14 15 19
``` |
| -5 | The data lines for a plane defined by three or more points (a best fit plane if more than three points) have the following format: |
| | ```
M  $3Dpppttttt.tttt
M  $3Diiijjj...zzz
...
``` |
	where the variables represent:
	`ppp` Number of points defining the line
	`ttttt.tttt` Deviation (Å), 0 if not used.
	`iii` Each `iii`, `jjj`, and `zzz` is the ID number jjj of an item in the model that defines the plane
	`jjj`
	`...`
	`zzz` (to maximum of 20 items per data line)
	The following plane is defined by the three points 1, 5, and 14. The first line is the identification line. The second and third lines are the data lines.
| | ```
M  $3D -5  4 PLANE_2
M  $3D  3    0.0000
M  $3D  1 5 14
``` |
| -6 | The data line for a plane defined by a point and a line has the following format: |
| | ```
M  $3Diiilll
``` |
	where the variables represent:
	`iii` ID number of a point
	`lll` ID number of a line
	The following plane is defined by the point 1 and the plane 16. The first line is the identification line. The second line is the data line.
| | ```
M  $3D -6  3 PLANE_1
M  $3D  1 16
``` |

-7	The data lines of a centroid defined by points have the following format:
| | ```
M  $3Dppp
M  $3Diiijjj...zzz ...
``` |
	where the variables represent:
	`ppp` Number of points defining the centroid
	`iii` Each `iii`, `jjj`, and `zzz` is the ID number `jjj` of an item in the model that defines the centroid
	`jjj`
	`...`
	`zzz` (maximum of 20 items per data line).
	The following centroid, `ARO_CENTER`, is defined by 3 items: 6, 8, and 10. The first line is the identification line. The second and third lines are the data lines.
| | ```
M  $3D -7  1 ARO_CENTER
M  $3D  3
M  $3D  6  8 10
``` |
| **-8** | The data line for a normal line defined by a point and a plane has the following format: |
| | ```
M  $3Diiijjj
``` |
	where the variables represent:
	`iii` ID number of a point
	`jjj` ID number of a plane
	The following normal line, `ARO_NORMAL`, is defined by the point 14 and the plane 15. The first line is the identification line. The second line is the data line.
| | ```
M  $3D -8  1 ARO_NORMAL
M  $3D 14 15
``` |
| **-9** | The data line for a distance defined by two points and a range (Å) has the following format: |
| | ```
M  $3Diiijjjddddd.ddddzzzzz.zzzz
``` |
	where the variables represent:
	`iii` ID number of a point
	`jjj` ID number of a second point
	`ddddd.dddd` Minimum distance (Å)
	`zzzzz.zzzz` Maximum distance (Å)
	The following distance, L, is between items 1 and 14 and has a minimum distance of 4.9Å and a maximum distance of 6.0Å. The first line is the identification line. The second line is the data line.
| | ```
M  $3D -9  6 L
M  $3D  1 14   4.9000    6.0000
``` |
| **-10** | The data line for a distance defined by a point, line, and a range (Å) has the format: |
| | ```
M  $3Diiilllddddd.ddddzzzzz.zzzz
``` |
	where the variables represent:
	`iii` ID number of a point
	`lll` ID number of a line
	`ddddd.dddd` Minimum distance (Å)
	`zzzzz.zzzz` Maximum distance (Å)

-11	The data line for a distance defined by a point, plane, and a range (Å) has the format:
	`M  $3Diiijjjddddd.ddddzzzzz.zzzz`
	where the variables represent:
	`iii` ID number of a point
	`jjj` ID number of a plane
	`ddddd.dddd` Minimum distance (Å)
	`zzzzz.zzzz` Maximum distance (Å)
-12	The data line for an angle defined by three points and a range (in degrees) has the following format:
	`M  $3Diiijjjkkkddddd.ddddzzzzz.zzzz`
	where the variables represent:
	`iii` ID number of a point
	`jjj` ID number of a second point
	`kkk` ID number of a third point
	`ddddd.dddd` Minimum degrees
	`zzzzz.zzzz` Maximum degrees
	The following angle, `THETA1`, is defined by the three points: 5, 17, and 16. The minimum angle is 80° and the maximum is 105°. The first line is the identification line. The second line is the data line.
	`M  $3D-12  5 THETA1` `M  $3D  5 17 16   80.0000  105.0000`
-13	The data line for an angle defined by two lines and a range (in degrees) has the following format:
	`M  $3Dlllmmmddddd.ddddzzzzz.zzzz`
	where the variables represent:
	`lll` ID number of a line, mmm ID number of a second line
	`ddddd.dddd` Minimum degrees
	`zzzzz.zzzz` Maximum degrees
	`THETA2` is defined by the lines 27 and 26 with maximum and minimum angles of 45° and 80°. The first line is the identification line. The second line is the data line.
	`M  $3D-13  5 THETA2` `M  $3D 27 26   45.0000   80.0000`
-14	The data line for an angle defined by two planes and a range (in degrees) has the following format:
	`M  $3Diiijjjddddd.ddddzzzzz.zzzz`
	where the variables represent:
	`iii` ID number of a plane
	`jjj` ID numbers of a second plane
	`ddddd.dddd` Minimum degrees
	`zzzzz.zzzz` Maximum degrees

-15	The data line for a dihedral angle defined by four points and a range (in degrees) has the following format:
	
	`M  $3Diiijjjkkklllddddd.ddddzzzzz.zzzz`
	
	where the variables represent:
	
	`iii` ID number of a point
	
	`jjj` ID number of a second point
	
	`kkk` ID number of a third point
	
	`lll` ID number of a fourth point
	
	`ddddd.dddd` Minimum degrees
	
	`zzzzz.zzzz` Maximum degrees
	
	`DIHED1` is defined by the items 7, 6, 4, and 8 with minimum and maximum angles of 45° and 80°, respectively. The first line is the identification line. The second line is the data line.
	
	`M  $3D-15  5 DIHED1`
	`M  $3D 7  6  4  8   45.0000   80.0000`

-16	The data lines for an exclusion sphere defined by a point and a distance (Å) have the following format:
	
	`M  $3Diiiuuuaaaddddd.dddd`
	`M  $3Dbbbccc...zzz ...`
	
	where the variables represent:
	
	`iii` ID number of the center of the sphere
	
	`uuu` 1 or 0. 1 means unconnected atoms are ignored within the exclusion sphere during a search; 0 otherwise
	
	`aaa` Number of allowed atoms
	
	`ddddd.dddd` Radius of sphere (Å)
	
	`bbb` Each `bbb`, `ccc`, and `zzz` is an ID number of an allowed atom
	
	`ccc`
	
	…
	
	`zzz` (to maximum of 20 items per data line)
	
	The following exclusion sphere is centered on point 24, has a radius of 5, and allows atom 9 within the sphere. The first line is the identification line. The second and third lines are the data lines.
	
	`M  $3D-16  7 EXCL_SPHERE`
	`M  $3D 24  0  1    5.0000`
	`M  $3D  9`

-17	The data lines of the fixed atoms have the following format:
	
	`M  $3Dppp`
	`M  $3Diiijjj...zzz ...`
	
	where the variables represent:
	
	ppp Number of fixed points
	
	`iii` Each `iii`, `jjj`, and `zzz` is an ID number of a fixed atom
	
	`jjj`
	
	`zzz` (to maximum of 20 items per data line)
	
	The following examples shows 4 fixed atoms. The first line is the identification line. The second and third lines are the data lines.
	
	`M  $3D-17`
	`M  $3D  4`
	`M  $3D  3  7 12 29`

3D data constraints [3D, Query]

A positive integer is used as a type identifier to indicate an atom or atom-pair data constraint. Two lines are used to describe a data constraint. The lines have the following format:

```
M  $3Dnnncccaaa...aaabbbbbbbbppppppppsss...sss
M  $3Diiijjjddd...ddd
```

where the variables represent:

nnn:	Database-field number
ccc:	Color
aaa...aaa:	Database-field name (up to 30 characters)
bbbbbbbb:	/BOX = box-number (source of data) (up to 8 characters)
pppppppp:	/DASP = n1, n2 where n1 and n2 are digits from 1-9 (data size and position) (up to 9 characters)
sss...sss:	/DISP = 3DN (name), 3DV (value), 3DQ (query), NOT (no text). First three in any combination to maximum total of 15 characters
iii:	ID number of an atom
jjj:	ID number of a second atom for atom-pair data, 0 if data is atom data
ddd...ddd:	Data constraint (based on format from database) (up to 64 characters)

ISIS 3D data query syntax and MACCS-II 3D data query syntax are not identical. The ISIS data query requires a search operator, a blank space, then one or more operands. For more information on ISIS data query syntax, see the ISIS Help system entries on SBF (Search By Form) or QB (Query Builder) for entering text in a query. For information on MACCS-II data searches, see the MACCS-II Command Language Reference.

Note: For MACCS-II, the atom number 999 stands for all atoms. The MACCS-II wild card character (@) can be used in the data constraints.

The following example shows a numeric data constraint for the field CNDO.CHARGE on atom 12. The first line is the identification line. The second line is the data line.

```
M  $3D  7  0 CNDO.CHARGE
M  $3D 12  0   -0.3300   -0.1300
```

The following example shows a numeric data constraint for the field BOND.LENGTH on the atom pair 1 and 4. The first line is the identification line. The second line is the data line.

```
M  $3D  9  0 BOND.LENGTH
M  $3D  1  4   2.0500   1.8200
```

The following example shows a data constraint allowing any charge value for the field CHARGE on all the atoms. The first line is the identification line. The second line is the data line.

```
M  $3D 12  0 CHARGE
M  $3D999  0 @
```

Chapter 3: Atom Limit Enhancements

The formats presented in this chapter were added to support the chemical representation enhancements of MDL ISIS Desktop 2.0.

Phantom Extra Atom

The format for phantom extra atom information is as follows:

```
M  PXA aaaxxxxx.xxxxyyyy.yyyyzzzzz.zzzz H   e...
```

where:

`aaa:`	Index of (real) atom for attachment
`xyz:`	Coordinates for the added atom
`H:`	Atom symbol
`e...:`	Additional text string (for example, the superatom label)

The bond to the added phantom atom is added as a crossing bond to the outermost Sgroup that contains atom aaa. Note this appendix supplies coordinates and up to 35 characters of 'label' that can be used for the ISIS/Desktop superatom conversion mechanism. ISIS/Desktop uses this appendix to register hydrogen-only superatoms, which are often used as superatom leaving groups on the desktop, but which cannot be directly registered into ISIS/Host databases. The hydrogen-only leaving groups are converted to PXA appendices for registration, and converted back when ISIS/Desktop reads the structure.

The following are limitations on phantom extra atom:

- Superatom nesting cases

- No bonded phantom atom-phantom atom support

Superatom Attachment Point

The format for superatom attachment point is as follows:

```
M  SAP sssnn6 iii ooo cc
```

where:

`sss:`	Index of superatom Sgroup
`nn6:`	Number of `iii`,`ooo`,`cc` entries on the line (6 maximum)
`iii:`	Index of the attachment point atom
`ooo:`	Index of atom in `sss` that leaves when `iii` is substituted
`cc:`	2 character attachment identifier (for example, `H` or `T` for head/tail). No validation of any kind is performed, and ' ' is allowed. ISIS/Desktop uses the first character as the ID of the leaving group to attach if the bond between `ooo` and `iii` is deleted, and uses the second character to indicate the sequence polarity: `l` for left, `r` for right, and `x` for none (a crosslink).

The bond `iii-ooo` is either a sequence bond, a sequence crosslink bond, or a bond to a leaving group that terminates a sequence or caps a crosslink bond. In some cases, this bond may have been deleted by the user, probably to perform a substructure search. In this case, `ooo` will be 0. If the leaving group attached to `iii` consists of only a hydrogen, the leaving group will be replaced by a Phantom Extra Atom, as previously described. In this case, `iii` is set equal to `ooo` as a signal to ISIS/Desktop that a hydrogen-only leaving group must be reattached to iii.

An attachment point entry is one `iii,ooo,cc` triad.

Multiple M SAP lines are permitted for each superatom Sgroup to the maximum of the atom attachment limit. The order of the attachment entries is significant because the first `iii,ooo,c` becomes the first connection made when drawing to the collapsed superatom, and so forth.

Superatom Class

The format for superatom class is as follows:

`M   SCL sss d...`

where:

`sss:`	Index of superatom Sgroup
`d...:`	Text string (for example, `PEPTIDE`, ...) 69 characters maximum

This appendix identifies the class of the superatom Sgroup. It distinguishes, for example, peptide groups from nucleotides.

Large REGNO

The format for the regno appendix is as follows:

`M   REG r...`

where:

`rrr:`	Free format integer regno

This appendix supports overflow of the I6 regno field in the molfile header. If this appendix is present, the value of the regno in the molfile header is superceded.

Sgroup Bracket Style

The format for the Sgroup bracket style is as follows:

`M   SBTnn8 sss ttt ...`

where:

`sss:`	Index of Sgroup
`ttt:`	Bracket display style: 0 = default, 1 = curved (parenthetic) brackets

This appendix supports altering the display style of the Sgroup brackets.

Chapter 4: Molfiles

Overview

A molfile consists of a header block and a connection table. The following shows a molfile for alanine corresponding to the following structure:

Figure 10 Molfile organization illustrated using alanine


```
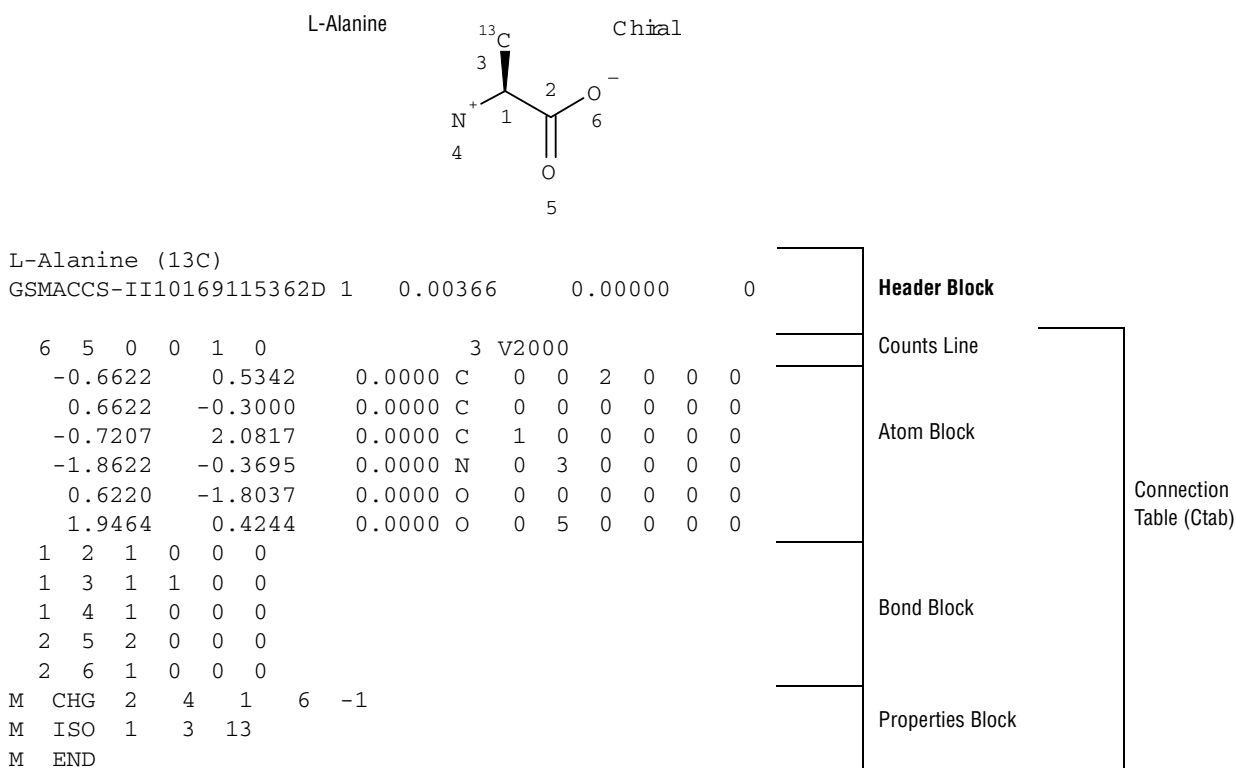L-Alanine (13C)
GSMACCS-II10169115362D 1   0.00366     0.00000      0

  6  5  0  0  1  0              3 V2000
   -0.6622    0.5342    0.0000 C   0  0  2  0  0  0
    0.6622   -0.3000    0.0000 C   0  0  0  0  0  0
   -0.7207    2.0817    0.0000 C   1  0  0  0  0  0
   -1.8622   -0.3695    0.0000 N   0  3  0  0  0  0
    0.6220   -1.8037    0.0000 O   0  0  0  0  0  0
    1.9464    0.4244    0.0000 O   0  5  0  0  0  0
  1  2  1  0  0  0
  1  3  1  1  0  0
  1  4  1  0  0  0
  2  5  2  0  0  0
  2  6  1  0  0  0
M  CHG  2   4   1   6  -1
M  ISO  1   3  13
M  END
```

Header Block

Counts Line

Atom Block

Bond Block

Properties Block

Connection Table (Ctab)

The format for a molfile is:

Header Block:	Identifies the molfile: molecule name, user's name, program, date, and other miscellaneous information and comments
Ctab Block:	Connection table (described in Chapter 2)

The detailed format for the header block follows.

The Header Block

Line 1:	Molecule name. This line is unformatted, but like all other lines in a molfile may not extend beyond column 80. If no name is available, a blank line must be present.
	Caution: This line must not contain any of the reserved tags that identify any of the other CTAB file types such as $MDL (RGfile), $$$$ (SDfile record separator), $RXN (rxnfile), or $RDFILE (RDfile headers).
Line 2:	This line has the format:
	``` IIPPPPPPPPMMDDYYHHmmddSSssssssssssEEEEEEEEEEEEERRRRR (FORTRAN: A2<--A8--><---A10-->A2I2<--F10.5-><---F12.5--><-I6->) ```
	User's first and last initials (l), program name (P), date/time (M/D/Y,H:m), dimensional codes (d), scaling factors (S, s), energy (E) if modeling program input, internal registry number (R) if input through MDL form.
	A blank line can be substituted for line 2.
	If the internal registry number is more than 6 digits long, it is stored in an M REG line (described in Chapter 3).
Line 3:	A line for comments. If no comment is entered, a blank line must be present.

Chapter 5: RGfiles

Overview

The format of an RGfile (Rgroup query file) is shown below. Lines beginning with $ define the overall structure of the Rgroup query; the molfile header block is embedded in the Rgroup header block.

In addition to the primary connection table (Ctab block) for the root structure, a Ctab block defines each member (*m) within each Rgroup (*r).

```
$MDL  REV 1 date/time
$MOL
$HDR
[Molfile Header Block (see Chapter 4) = name, pgm info, comment]
$END HDR
$CTAB
[Ctab Block (see Chapter 2) = count + atoms + bonds + lists + props]
$END CTAB
$RGP
rrr    [where rrr = Rgroup number]
$CTAB
[Ctab Block]
$END CTAB
$END RGP
$END MOL
```

where:

*r (Rgroup) is repeated. MACCS-II and ISIS/Desktop have an internal limit of 32 Rgroups.

*m （member）is repeated. MACCS-II has a maximum internal limit of 255 total atoms and bonds per Rgroup.

Figure 11   Example of an RGfile (Rgroup query file)

```
$MDL  REV  1 16OCT91  15:40
$MOL
$HDR
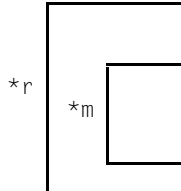
GSMACCS-II10169115402D 1   0.00353      0.00000      0

$END HDR
$CTAB
  9  9  0  0  0  0              4 V2000
    1.3337    0.7700    0.0000 C   0   0   0   0   0   0
       .          .                      .
    1.3337   -0.7700    0.0000 C   0   0   0   0   0   0
    0.0000    3.0800    0.0000 R#  0   0   0   0   0   0
    2.6674    1.5400    0.0000 R#  0   0   0   0   0   0
   -2.6674    1.5400    0.0000 R#  0   0   0   0   0   0
  1  2  1  0  0  0
       .          .
  3  9  1  0  0  0
M  RGP  3    7   1   8   2   9   2
M  LOG  1    1   2   0
M  LOG  1    2   0   0   0
M  END
$END CTAB
$RGP
   1
$CTAB
  1  0  0  0  0  0              2 V2000
   12.2100   14.3903    0.0000 C   0   0   0   0   0   0
M  APO  1    1   1
M  END
$END CTAB
$END RGP
$RGP
   2
$CTAB
  2  1  0  0  0  0              2 V2000
   -1.4969    0.0508    0.0000 C   0   0   0   0   0   0
    0.0431    0.0508    0.0000 O   0   0   0   0   0   0
  1  2  2  0  0  0
M  APO  1    1   1
M  END
$END CTAB
$CTAB
  1  0  0  0  0  0              2 V2000
   12.2100   14.3903    0.0000 N   0   0   0   0   0   0
M  APO  1    1   1
M  END
$END CTAB
$END RGP
$END MOL
```

Molfile header block

RGfile header block

Counts line

Atom block of root

Number of entries on line

Rgroup label location and Rgroup number

Bond block of root
Atom list block
Stext block
Properties block of root

Ctab for Rgroup root

If/then, RestH, and Occurrence

Ctab for Rgroup 1 member 1

Block for Rgroup R1

Ctab for Rgroup 2 member 1

Ctab for Rgroup 2 member 2

Block for Rgroup R2

The RGfile shown in corresponds to the following R group query:

Chapter 6: SDfiles

Overview

An SDfile (structure-data file) contains the structural information and associated data items for one or more compounds. The format is:

```
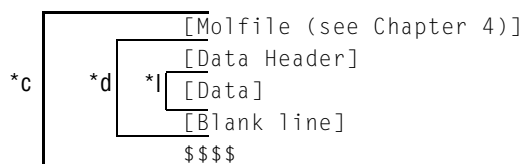           [Molfile (see Chapter 4)]
           [Data Header]
*c   *d  *l [Data]
           [Blank line]
           $$$$
```

where:

> *l is repeated for each line of data
> *d is repeated for each data item
> *c is repeated for each compound

A *[Molfile]* block has the molfile format described in Chapter 4.

A *[Data Header]* (one line) precedes each item of data, starts with a *greater than* (>) sign, and contains at least one of the following:

- The field name enclosed in angle brackets. For example: `<melting.point>`

- The field number, `DTn` , where *n* represents the number assigned to the field in a MACCS-II database

Optional information for the data header includes:

- The compound's external and internal registry numbers. External registry numbers must be enclosed in parentheses.

- Any combination of information

The following are examples of valid data headers:

```
> <MELTING.POINT>
> 55     (MD-08974)     <BOILING.POINT>    DT12
> DT12    55
> (MD-0894)   <BOILING.POINT>   FROM ARCHIVES
```

Figure 12   Example of an SDfile

```
1,2 CYCLO-C6 DI-COOH TRANS,L                                    ⎤
  MACCS-II060390162920162D 1   0.00339     0.00000    25       ⎦ Header block
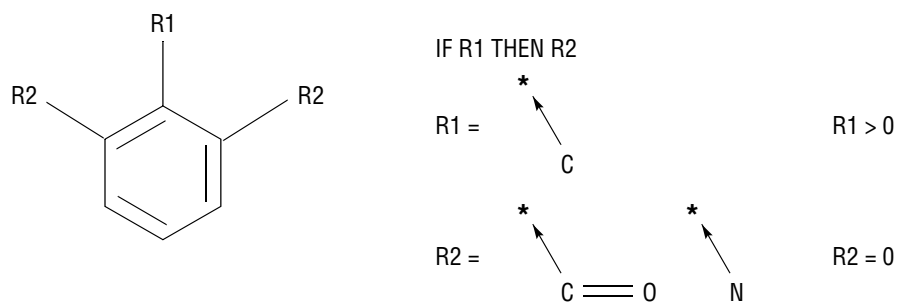
 12 12  0  0  1  0              1 V2000                          ⎤
   -0.0238   -0.7702    0.0000 C   0  0  1  0  0  0             │
            .                        .                          │ Connection
    2.6974    0.7634    0.0000 O   0  0  0  0  0  0             │ table
  1  2  1  0  0  0                                              │
         .        .                                             │
  7 10  1  0  0  0                                              │
M  END                                                          ⎦
> 25  <MELTING.POINT>        ⎤ Data header                      ⎤
179.0 - 183.0               ⎥ Data                              │ Data item
                            ⎦ Blank line                        │
> 25  <DESCRIPTION>                                             │
PW(W)                                                           │
                                                                │
> 25  <ALTERNATE.NAMES>                                         │
1,2 CYCLOHEXANE-DICARBOXYLIC ACID TRANS,L                       │ Data items
HEXAHYDROPHTHALIC ACID TRANS,L                                  │
                                                                │
> 25  <DATE>                                                    │
01-10-1980                                                      │
                                                                │
> 25  <CRC.NUMBER>                                              │
C-0710Dat                                                       ⎦

$$$$                                                              Delimiter
2-METHYL FURAN                                                  ⎤
  MACCS-II060390163020163D 1   0.00186     0.00000    29       ⎦ Header block

  6  6  0  0  0  0              1 V2000                          ⎤
    0.5343    0.3006    0.0000 C   0  0  0  0  0  0             │
            .                        .                          │ Connection
   -2.0038    0.2857    0.0000 C   0  0  0  0  0  0             │ table
  1  2  2  0  0  0                                              │
         .        .                                             │
  5  6  2  0  0  0                                              │
M  END                                                          ⎦
> 29  <DENSITY>                                                 ⎤
0.9132 - 20.0                                                   │
                                                                │
> 29  <BOILING.POINT>        ⎤ Data header                      │
63.0 (737 MM)               ⎥ Data                              │
79.0 (42 MM)                ⎦ Blank line                        │ Data items
                                                                │
> 29  <ALTERNATE.NAMES>                                         │
SYLVAN                                                          │
                                                                │
> 29  <DATE>                                                    │
09-23-1980                                                      │
                                                                │
> 29  <CRC.NUMBER>                                              │
F-0213                                                          ⎦

$$$$                                                              Delimiter
```

Header block · *Connection table* · *Molfile* · *Data item* · *Data items* · *Non-structural data* · *Compound*

A *[Data]* value may extend over multiple lines containing up to 200 characters each. A blank line terminates each data item.

A line beginning with four dollar signs ($$$$) terminates each complete data block describing a compound.

A datfile (data file) is effectively an SDfile with no *[Molfile]* descriptions or $$$$ delimiters. The *[Data Header]* in a datfile must include either an external or internal registry number in addition to a field name or number.

Notes about using blank lines

- *Only* one blank line should terminate a data item.

- There should *only* be one blank line between the last data item and the $$$$ delimiter line.

- If the SDfile only contains structures, there can be *no* blank line between the last "M END" and the $$$$ delimiter line.

SDfile after a CFS search

After a conformationally flexible substructure (CFS) search, the following format information is appended by MDL ISIS/Base PL to your SDfile after the connection table:

- Query information (M $3D appendix lines added to embedded molfile)

- CFS generated data (*DATA)

- MAPPED ATOMS and BONDS

This information describes, for example, how query atoms are mapped, the atom coordinates in models, and what is fitted during a CFS search.

Figure 13   Example of SDfile with appended CFS query information

```
M   CHG   2   14   -1   16    1
M   $3D   5                                  ┐
M   $3D -9   3                               │
M   $3D 13 18     6.3000     8.3000          │
M   $3D -9   3                               │
M   $3D 18   9    3.1000     5.1000          │
M   $3D -9   3                               ├  3D Query Fields
M   $3D 18   4    2.4000     4.4000          │
M   $3D -9   3                               │
M   $3D 13   9    2.8000     4.8000          │
M   $3D -9   3                               │
M   $3D 13   4    3.1000     5.1000          ┘
M   END
> 31 <*DATA>                                 ┐  CFS-Generated Data
Method = Derivative                          ┘

> 31 <MAPPED ATOMS AND BONDS>                ┐  Mapping Atoms and Bonds
(8 13 14 3 9 4 18) (12 13 7 8)               │
                                             ┘
$$$$
```

Chapter 7: Rxnfiles

Overview

Rxnfiles contain structural data for the reactants and products of a reaction. To see a sample rxnfile go to . The format is:

```
            [Rxnfile Header]
            rrrppp
 *r         $MOL
            [Molfile of reactant]
 *p         $MOL
            [Molfile of product]
```

where:

*r is repeated for each reactant
*p is repeated for each product

Header Block

Line 1:	$RXN in the first position on this line identifies the file as a reaction file.
Line 2:	A line for the reaction name. If no name is available, a blank line must be present.

Line 3:	User's initials (I), program name and version (P), date/time (M/D/Y, H:m), and reaction registry number (R). This line has the format:
	
	`                IIIIIIPPPPPPPPPPMMDDYYYYHHmmRRRRRRR`
	`(FORTRAN:   <-A6-><---A9--><---A12----><--I7-> )`
	
	A blank line can be substituted for line 3.
	
	If the internal registry number is more than 7 digits long, it is stored in an "M REG" line (described in Chapter 3).
	
	Note: In rxnfiles produced by earlier versions of MDL ISIS/Host, the year occupied two digits instead of four. There are corresponding minor changes in the adjacent fields. The format of the line is:
	
	`                IIIIIIPPPPPPPPPPMMDDYYHHmmRRRRRRRR`
	`(FORTRAN:   <-A6-><---A10--><---A10--><--I8--> )`

Line 4	A line for comments. If no comment is entered, a blank line must be present.

Figure 14   Rxnfile for the acylation of benzene

```
$RXN

    REACCS81  1017911041    7439

  2  1
$MOL

  REACCS81101791104I2D 1   0.00380     0.00000   315

  4  3  0  0  0  0              1 V2000
    0.3929   -0.2577    0.0000 C   0  0  0  0  0  0  0  0  0  1  0  0
   -1.0590   -0.7710    0.0000 C   0  0  0  0  0  0  0  0  0  2  0  0
    0.3929    1.2823    0.0000 O   0  0  0  0  0  0  0  0  0  3  0  0
    1.6503   -1.1468    0.0000 Cl  0  0  0  0  0  0  0  0  0  0  0  0
  1  2  1  0  0  0  2
  1  3  2  0  0  0  2
  1  4  1  0  0  0  4
M  END
$MOL

  REACCS81101791104I2D 1   0.00371     0.00000     8

  6  6  0  0  0  0              1 V2000
    1.3335   -0.7689    0.0000 C   0  0  0  0  0  0  0  0  0  5  0  0
    1.3335    0.7689    0.0000 C   0  0  0  0  0  0  0  0  0  6  0  0
    0.0000   -1.5415    0.0000 C   0  0  0  0  0  0  0  0  0  7  0  0
    0.0000    1.5415    0.0000 C   0  0  0  0  0  0  0  0  0  8  0  0
   -1.3335   -0.7689    0.0000 C   0  0  0  0  0  0  0  0  0  9  0  0
   -1.3335    0.7689    0.0000 C   0  0  0  0  0  0  0  0  0 10  0  0
  1  2  1  0  0  0  2
  1  3  2  0  0  0  2
  2  4  2  0  0  0  2
  3  5  1  0  0  0  2
  4  6  1  0  0  0  2
  5  6  2  0  0  0  2
M  END
$MOL

  REACCS81101791104I2D 1   0.00374     0.00000   255

  9  9  0  0  0  0              1 V2000
   -0.5311   -0.1384    0.0000 C   0  0  0  0  0  0  0  0  0  5  0  0
   -1.8626    0.6321    0.0000 C   0  0  0  0  0  0  0  0  0  6  0  0
   -0.5311   -1.6943    0.0000 C   0  0  0  0  0  0  0  0  0  7  0  0
    0.8191    0.6284    0.0000 C   0  0  0  0  0  0  0  0  0  1  0  0
   -3.2278   -0.1346    0.0000 C   0  0  0  0  0  0  0  0  0  8  0  0
   -1.8813   -2.4723    0.0000 C   0  0  0  0  0  0  0  0  0  9  0  0
    2.1282   -0.1085    0.0000 C   0  0  0  0  0  0  0  0  0  2  0  0
    0.8191    2.2292    0.0000 O   0  0  0  0  0  0  0  0  0  3  0  0
   -3.2278   -1.6831    0.0000 C   0  0  0  0  0  0  0  0  0 10  0  0
  1  2  1  0  0  0  2
  1  3  2  0  0  0  2
  1  4  1  0  0  0  4
  2  5  2  0  0  0  2
  3  6  1  0  0  0  2
  4  7  1  0  0  0  2
  4  8  2  0  0  0  2
  5  9  1  0  0  0  2
  6  9  2  0  0  0  2
M  END
```

Header block

#Reactants and #Products

Molfile delimiter

Molfile for first reactant

Molfile delimiter

Molfile for second reactant

Molfile delimiter

Molfile for product

Reactants/Products

A line identifying the number of reactants and products, in that order. The format is:

`rrrppp`

where the variables represent:

   `rrr`   Number of reactants*

   `ppp`   Number of products*

* MDL ISIS/Host has a limit of 8 reactants and 8 products. MDL Direct does not impose any limits.

Molfile Blocks

A series of blocks, each starting with `$MOL` as a delimiter, giving the molfile for each reactant and product in turn. The molfile blocks are always in the same order as the molecules in the reaction; reactants first and products second.

The rxnfile in corresponds to the following reaction:

Chapter 8: RDfiles

Overview

An RDfile (reaction-data file) consists of a set of editable "records." Each record defines a molecule or reaction, and its associated data. An example RDfile incorporating the rxnfile described in Chapter 7 is shown later in this chapter (see). The format for an RD file is:

```
                    [RDfile Header]
                    [Molecule or Reaction Identifier]
       *r      *d  [Data-field Identifier]
                    [Data]
```

where:

> *d is repeated for each data item
> *r is repeated for each reaction or molecule

Each logical line in an RDfile starts with a keyword in column 1 of a physical line. One or more blanks separate the first argument (if any) from the keyword. The blanks are ignored when the line is read. After the first argument, blanks are significant.

An argument longer than 80 characters breaks at column 80 and continues in column 1 of the next line. (The argument may continue on additional lines up to the physical limits on text length imposed by the database.)

The RDfile must not contain any blank lines except as part of embedded molfiles, rxnfiles, or data. An identifier separates records.

RDfile Header

Line 1:	$RDFILE 1: The *[RDfile Header]* must occur at the beginning of the physical file and identifies the file as an RDfile. The version stamp "1" is intended for future expansion of the format.
Line 2:	$DATM: Date/time (M/D/Y, c) stamp. This line is treated as a comment and ignored when the program is read.

Molecule and Reaction Identifiers

A *[Molecule or Reaction Identifier]* defines the start of each complete record in an RDfile. The form of a *molecule* identifier must be one of the following:

```
$MFMT [$MIREG internal-regno [$MEREG external-regno]] embedded molfile
$MIREG internal-regno
$MEREG external-regno
```

where:

- $MFMT defines a molecule by specifying its connection table as a molfile

- $MIREG *internal-regno* is the internal registry number (sequence number in the database) of the molecule

- $MEREG *external-regno* is the external registry number of the molecule (any uniquely identifying character string known to the database, for example, CAS number)

- Square brackets ([]) enclose optional parameters

- An embedded molfile (see Chapter 4) follows immediately after the $MFMT line

The forms of a *reaction* identifier closely parallel that of a molecule:

```
$RFMT [$RIREG internal-regno [$REREG external-regno]] embedded rxnfile
$RIREG internal-regno
$REREG external-regno
```

where:

- $RFMT defines a reaction by specifying its description as a rxnfile

- $RIREG *internal-regno* is the internal registry number (sequence number in the database) of the reaction

- $REREG *external-regno* is the external registry number of the reaction (any uniquely identifying character string known to the database)

- Square brackets ([]) enclose optional parameters

- An embedded rxnfile (see Chapter 7) follows immediately after the $RFMT line

Data-field Identifier

The *[Data-field Identifier]* specifies the name of a data field in the database. The format is:

```
$DTYPE field name
```

Data

Data associated with a field follows the field name on the next line and has the form:

```
$DATUM datum
```

The format of *datum* depends upon the data type of the field as defined in the database. For example: integer, real number, real range, text, molecule regno.

For fields whose data type is "molecule regno," the *datum* must specify a molecule and, with the exception noted below, use one of the formats defined above for a molecular identifier. For example:

```
$DATUM $MFMT embedded molfile
$DATUM $MEREG external-regno
$DATUM $MIREG internal-regno
```

In addition, the following special format is accepted:

```
$DATUM molecule-identifier
```

Here, *molecule-identifier* acts in the same way as *external-regno* in that it can be any text string known to the database that uniquely identifies a molecule. (It is usually associated with a data field different from the *external-regno*.)

Figure 15 Example of a reaction RDfile

```
$RDFILE 1                                                           RDfile header
$DATM    10/17/91 10:41
$RFMT $RIREG 7439
$RXN                                                           Rxnfile
                                                               header
  REACCS81   1017911041     7439

  2  1                                                       #Reactants and #Products
$MOL
                                                             Molfile for
  REACCS8110179110412D 1   0.00380     0.00000    315        first
                                                             reactant
  4  3  0  0  0  0  0  0  0  0  0                                           First Rxn
     .     .    .                                                          record
  1  4  1  0  0  0  4
$MOL
                                                             Molfile for
  REACCS8110179110412D 1   0.00371     0.00000      8        second
                                                             reactant
  6  6  0  0  0  0  0  0  0  0  0
     .     .    .
  5  6  2  0  0  0  2
$MOL
                                                             Molfile for
  REACCS8110179110412D 1   0.00374     0.00000    255        product

  9  9  0  0  0  0  0  0  0  0  0
     .     .    .
  6  9  2  0  0  0  2
$DTYPE rxn:VARIATION(1):rxnTEXT(1)
$DATUM CrCl3
$DTYPE rxn:VARIATION(1):LITTEXT(1)
$DATUM A G Repin, Y Y Makarov-Zemlyanskii, Zur Russ Fiz-Chim, 44,
p.2360, 1974
           .          .          .         .          .       Data block
$DTYPE rxn:VARIATION(1):CATALYST(1):REGNO                      for reaction
$DATUM $MFMT $MIREG 688

  REACCS8110179110412D 1   0.00371     0.00000      0

  4  3  0  0  0  0  0  0  0  0  0
     .     .    .
  1  4  1  0  0  0  0
$DTYPE rxn:VARIATION(1):PRODUCT(1):YIELD
$DATUM 70.0
     .     .    .
$RFMT $RIREG 8410
$RXN                                                           Start of next Rxn
                                                               record
      REACCS81   1017911041    8410

  2  1
$MOL
...
```

Chapter 9: XDfiles

Overview

An XDfile (XML-data file) uses a standard set of XML elements that represent records of data. The XDfile format also provides:

- Metadata or information about the origin of the data. Unlike the other CTfile formats such as the SDfile or RDfile, the XDfile enables the consumer of the data to correctly interpret it.

- The ability to handle generalized data models, such as multiple structures and nonstructure fields per record, multiple reactions per record, multiline data, and binary data. None of the other CTfile formats such as SDFiles or RDFiles have this ability.

- Very few restrictions on data formatting within the actual content. Data formatting is based on XML, which does not have restrictions on line length or blank lines. See "Data Formatting" on page 54.

- Fast and easy parsing by using any XML parser. The XML data can be validated by using a DTD (Document Type Definition) or an XML schema that defines primitive rules which the data must follow. You can download the DTD and XML schema from the CTFile Formats download page at the Elsevier MDL web site (www.mdl.com). From the Elsevier MDL web site, click **Downloads > Download Products > Select a Product > MDL CTFile Formats no-fee > MDL CTFile Formats**.

  Note: Due to limitations in the capabilities of DTDs and XML schemas, an XDfile that is validated by the DTD and XML schema is not guaranteed to be correct. For example, it is possible to create a record with duplicate field values. The DTD and XML schema only provide a low level of validation. Applications which process XDfiles must provide high level checks on the consistency of the data.

- Flexibility in creating application-specific XML tags. Note, however, that it is the responsibility of the client application to interpret these custom tags.

The Data Source Service of MDL® Core Interface can read an XDfile and load it into an XML data source. It also provides tools for converting SDfiles and RDfiles to XDfiles. For information about these tools, see the "Utilities" section in the "Data Source Service" chapter of *MDL Core Interface Developer's Guide.*

Note: Although it is possible to convert SDfiles and RDfiles to XDfiles, there are risks involved in converting an XDfile to the original format. This is because the backward conversion will lose the metadata which is not supported in the other CTfile formats.

The rest of this chapter contains reference information about the XML elements in an XDfile. For the structure of an XDfile, see "Hierarchy of Elements" on page 55. For an alphabetic list of elements in an XDfile, see "Alphabetic List of Elements" on page 57.

Data Formatting

XML puts little or no restrictions on data formatting within the actual content. Data that is sensitive to white space, molfiles and rxnfiles in particular, must be enclosed in a CDATA section. You must set the `xml:space` attribute to `"preserve"`. The following example is a `Field` element that contains a molfile:

```
<Field name="Mol" xml:space="preserve"><![CDATA[
  -ISIS-  03190310282D

  1  0  0  0  0  0  0  0  0  0999 V2000
   -2.1000   -0.1208    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
M  END
]]></Field>
```

Note that for data outside a CDATA section, certain characters must be escaped as described in the following table:

Character	Representation in XML
<	<
>	>
"	"
'	'
&	&

For example, "`mp < 100`" becomes "`mp < 100`". See also the XML specification for more details.

Hierarchy of Elements

The following shows the hierarchy of elements within an XDfile. The attributes of the elements are not shown in order to more clearly show the structure of the XML:

```
<XDfile>
   <Dataset>
      <Source>
          <DataSource />
          <ProgramSource />
          <CreatorName />
          <CreateDate />
          <CreateTime />
          <Description />
          <Copyright />
      </Source>
      <Metadata>
          <FieldDef />
          <ParentDef>
             <FieldDef />
          </ParentDef>
      </Metadata>
      <Data>
          <Record>
             <Field>
             </Field>
             <Parent>
                <Record>
                   <Field>
                   </Field>
                </Record>
             </Parent>
          </Record>
      </Data>
   </Dataset>
</XDfile>
```

The following XML schema diagram also illustrates the hierarchy of elements in an XDfile:

Alphabetic List of Elements

Element	Parent	Brief Description
Copyright	Source	A copyright notice for the data
CreateDate	Source	The date the data was created
CreateTime	Source	The time the data was created
CreatorName	Source	The person who created the data
Data	Dataset	Contains records of data
Dataset	XDfile	Contains a collection of data
DataSource	Source	The source of the data
Description	Source	A description or comment about the data
Field	Record	The value of a field in a record
FieldDef	Metadata ParentDef	The definition of a field in the data
XDfile		The root element
Metadata	Dataset	Contains definitions of fields in the data
Parent	Record	Contains subrecords of data
ParentDef	Metadata	The definition of a parent field in the data
ProgramSource	Source	The program that created the data
Record	Data Parent	The unit of data; contains a set of field values
Source	Dataset	Contains information about the source of data

XDfile

The root element. XML data that uses the XDfile format must begin with `<XDfile>` and end with `</XDfile>`. It contains one or more <u>Dataset</u> elements.

Attributes

The data type of all attributes is `CDATA`.

Attribute	Required	Description
xmlns	Yes	The default namespace for elements in the XDfile. The default is `http://www.mdl.com/XDfile/NS`.
version	Yes	The version number of the XDfile data. The default is 1.2.

Parent Element

None

Child Elements

Element	Required	Description
<u>Dataset</u>	Yes	A single collection of data

Example

```xml
<?xml version="1.0"?>
<XDfile xmlns="http://www.mdl.com/XDfile/NS">
   <Dataset
       name="MySource">
   <Source>
      <DataSource>ACD99.1 Hview</DataSource>
      <ProgramSource>GCS 1.0</ProgramSource>
      <CreatorName>John Doe</CreatorName>
      <CreateDate DATEFORMAT="M/D/Y">7/22/99</CreateDate>
      <CreateTime TIMEFORMAT="24">13:45</CreateTime>
   </Source>
   <Metadata>
      <FieldDef name="Molstructure" type="Structure"
          molFormat="Chime"/>
      <FieldDef name="Reaction Vessel ID" type="FixedText"
          maxLength="15"/>
      <FieldDef name="Library ID" type="FixedText"
          maxLength="8"/>
      <FieldDef name="Tag Id" type="fixedText"
          maxLength="3"/>
      <ParentDef name="XYZ Test Results">
         <FieldDef name="Dose" type="Double"/>
         <FieldDef name="Response" type="Double"/>
      </ParentDef>
    </Metadata>
    <Data>
      <Record>
         <Field name="Molstructure"  xml:space="preserve">
           <![CDATA[7YALEn$Akl$QPbas35quasdfsdf38...]]></Field>
         <Field name="Reaction Vessel ID">SAR6077R-01A02</Field>
         <Field name="Library ID">SAR6077R</Field>
         <Field name="Tag Id">yes</Field>
         <Parent name="XYZ Test Results" totalRecords="2">
            <Record>
               <Field name="Dose">1.0</Field>
               <Field name="Response">99.0</Field>
            </Record>
            <Record>
               <Field name="Dose">2.0</Field>
               <Field name="Response">999.0</Field>
            </Record>
         </Parent>
      </Record>
      <Record>
         <Field name="Molstructure"  xml:space="preserve">
           <![CDATA[7YALEn$Akl$QPbas35quasdfsdf38...]]></Field>
         <Field name="Reaction Vessel ID">SAR6077R-01A03</Field>
         <Field name="Library ID">SAR6077R</Field>
         <Field name="Tag Id">no</Field>
      </Record>
    </Data>
   </Dataset>
</XDfile>
```

See Also

Dataset

A self-contained, single collection of data.

Attributes

The data type of all attributes is CDATA.

Attribute	Required	Description
name	Yes, if multiple datasets	The name of the dataset. If XDfile contains only one dataset, name is optional. If XDfile contains multiple datasets, name is required in order to identify each dataset.

Parent Element

XDfile

Child Elements

Element	Required	Description
Source	No	Describes the source of the data
Metadata	No	Describes the individual fields in the actual data
Data	Yes	The actual data

Example

```
<Dataset>
   <Metadata>
      <FieldDef name="CTAB" type="Structure" molFormat="Chime"/>
      <FieldDef name="Regno" type="Integer" isPrimaryKey="true"/>
      <FieldDef name="Molname" type="FixedText"/>
      <FieldDef name="Date" type="Date"/>
      <FieldDef name="DoubleValue" type="Double"/>
   </Metadata>
   <Data totalRecords="1">
      <Record>
         <Field name="CTAB">40Aieg6Mprlczdb^fg37JiOh</Field>
         <Field name="Regno">1</Field>
         <Field name="Molname">Test Structure</Field>
         <Field name="Date">05/10/2002</Field>
      </Record>
   </Data>
</Dataset>
```

See Also

Source

Contains information that describes the source of the data. Note that although each child element is optional, the order of the child elements is fixed. If a child element does not follow the order specified below, the element is discarded.

Attributes

None

Parent Element

Dataset

Child Elements

Element	Required	Description
DataSource	No	The source of the data.
ProgramSource	No	The program that created the data.
CreatorName	No	The name of the person who created the data.
CreateDate	No	The date the data was created. The default format is M-D-Y.
CreateTime	No	The time the data was created. The default format is 24-hour.
Description	No	A description or comments about the data.
Copyright	No	A copyright notice for the data.

Example

```
<Source>
    <DataSource>ACD99.1 Hview</DataSource>
    <ProgramSource>Core Interface 1.1</ProgramSource>
    <CreatorName>John Doe</CreatorName>
    <CreateDate dateOrder="M/D/Y">7/22/99</CreateDate>
    <CreateTime timeFormat="24">13:45</CreateTime>
</Source>
```

See Also

"Alphabetic List of Elements" on page 57, "Hierarchy of Elements" on page 55

DataSource

The source of the actual data.

Attributes

None

Parent Element

Source

Child Elements

None

Example

<DataSource>ACD99.1 Hview</DataSource>

See Also

ProgramSource

The program that created the actual data.

Attributes

None

Parent Element

[Source](#)

Child Elements

None

Example

```
<ProgramSource>MDL Core Interface 1.0</ProgramSource>
```

See Also

CreatorName

The person who created the actual data.

Attributes

None

Parent Element

[Source](#)

Child Elements

None

Example

```
<CreatorName>John Doe</CreatorName>
```

See Also

CreateDate

The date the actual data was created. The default format is M-D-Y.

Attributes

The data type of all attributes is CDATA.

Attribute	Required	Description
dateOrder	No	The format for the date in `<CreateDate>`. The separator character is specified as part of the format. The valid formats are: M-D-Y D-M-Y Y-M-D Y-D-M The default format is M-D-Y.

Parent Element

Source

Child Elements

None

Example

```
<CreateDate dateOrder="M/D/Y">7/22/99</CreateDate>
```

See Also

CreateTime

The time the actual data was created. The default format is 24-hour.

Attributes

The data type of all attributes is CDATA.

Attribute	Required	Description
timeFormat	No	The format for the time in `<CreateTime>`. The valid values are: 24 12 The default format is 24.

Parent Element

[Source](#)

Child Elements

None

Example

```
<CreateTime>24:15</CreateTime>
```

See Also

Description

A description or comment about the actual data.

Attributes

None

Parent Element

[Source](#)

Child Elements

None

Example

<Description>This is test data.</Description>

See Also

["Alphabetic List of Elements" on page 57](#), ["Hierarchy of Elements" on page 55](#)

Copyright

A copyright notice for the data.

Attributes

None

Parent Element

[Source](#)

Child Elements

None

Example

<Copyright>Copyright 2003 by Acme Corporation </Copyright>

See Also

Metadata

Contains information that describes the fields in the data. Applications can use this information to allow automatic creation of database tables or better formatting of data. The hierarchical structure of the `Metadata` element must match the data model contained in the [Data] element.

Attributes

None

Parent Element

[Dataset]

Child Elements

Element	Required	Description
ParentDef	No	Contains information about a single parent within the data
FieldDef	No	Contains information about a single field within the data

Example

```
<Metadata>
   <FieldDef name="Molstructure"
        type="Structure"
        molFormat="Chime"/>
   <FieldDef name="Reaction Vessel ID"
        type="FixedText"
        maxLength="15"/>
   <FieldDef name="Library ID"
        type="FixedText"
        maxLength="8"/>
   <FieldDef name="Tag Id"
        type="FixedText"/>
   <ParentDef name="XYZ Test Results">
      <FieldDef name="Dose"
           type="Double"/>
      <FieldDef name="Response"
           type="Double"/>
   </ParentDef>
</Metadata>
```

See Also

ParentDef

Contains information about a single parent field within the data.

Note: Do not create unnecessary parent fields at the root. Parent fields should only be used when there is an actual hierarchical structure to the data. For example, an SDFile would not have `ParentDef` elements, but would only have multiple `FieldDef` elements directly under the `Metadata` element. The actual data in the SDFile follows this format by having multiple `Record` elements directly under the `Data` element.

Attributes

The data type of all attributes is `CDATA`.

Attribute	Required	Description
name	Yes	The name of the parent field. This is a simple name because the context of the parent in the hierarchy is implied by the XML structure.

Parent Element

Metadata

ParentDef

Child Elements

Element	Required	Description
FieldDef	Yes	Contains information about a single field within the data
ParentDef	No	Contains information about a single parent within the data. A parent within a parent creates a hierarchy.

Example

```
<ParentDef name="XYZ Test Results">
   <FieldDef name="Dose"
        type="Double"/>
   <FieldDef name="Response"
        type="Double"/>
</ParentDef>
```

See Also

FieldDef

Contains information about a single field within the data.

Attributes

The data type of all attributes is `CDATA`.

Attribute	Required	Description
`name`	Yes	The name of the field. This is a simple name because the context of the field in the hierarchy is implied by the XML structure.

`type`	Yes	The type of data in the field. Applications can define their own type, but it will be the responsibility of client applications to interpret these. The valid values are:
		
		`Reaction`
		
		`Structure`
		
		`Integer`
		
		`Double`
		
		`FixedText`
		
		`VariableText`
		
		`Date`
		
		`Time`
		
		`DateTime`
		
		`Binary`

`isKey`	No	"`true`" if the field is a key. The valid values are:
		`true`
		`false`
		The default value is `false`.

`isPrimaryKey`	No	"`true`" if the field is a primary key. The valid values are:
		`true`
		`false`
		The default value is `false`.

`nativeName`	No	The name of the field specific to the source from which it is derived. For example, a native name for an Oracle database is *schema.table.column*.

`encoding`	No	The encoding set used by the data. Sample values are:
		`binhex`
		`rot64`

Attribute	Required	Description
charset	No	The character set used by the text. A sample value is `ISO-8859-1`.

decimalSeparator	No	The decimal separator used for floating point numbers. The valid values are:
		
		`Period`
		
		`Comma`
		
		The default value is `Period`.

maxLength	No	The maximum length of the data in this field. For binary fields in a relational data source (such as `CLOB` and `BLOB` fields), the maximum length varies depending on the version of the Oracle JDBC driver.

molFormat	No	The format for the structure, if the field contains structures. The valid values are:
		
		`Molfile`
		
		`Chime`
		
		`Smiles`
		
		The default value is `Molfile`.

rxnFormat	No	The format for the reaction, if the field contains reactions. The valid values are:
		
		`Rxnfile`
		
		`Chime`
		
		The default value is `Rxnfile`.

molVersion	No	The version of the data format. `molVersion` is currently only applicable to structures. The absence of this attribute implies a pre-V2000 molfile. The valid values are:
		
		`V2000`
		
		`V3000`

dateOrder	No	The order and format of a date field. The value specifies the separator character. Dates must be specified using numbers only; do not use month names. The valid orders are:
		
		`M-D-Y`
		
		`D-M-Y`
		
		`Y-M-D`
		
		`Y-D-M`
		
		The default value is `M-D-Y`.

Attribute	Required	Description
`timeOrder`	No	The order of a date field. The value specifies the separator character. The valid orders are: `h:m` `m:h` The default value is `h:m`.
`timeFormat`	No	The format of a time field. The valid values are: `24` `12` The default value is `24`.
`precision`	No	The precision for a real number field. This is the total number of digits.
`scale`	No	The scale for a real number field. This is the number of digits to the right of the decimal point.
`nullsAllowed`	No	"`true`" if the field can contain nulls. The valid values are: `true` `false` The default value is `false`.
`isIndexed`	No	"`true`" if the field is indexed. The valid values are: `true` `false` The default value is `false`.
`isHidden`	No	"`true`" if the field is hidden. The valid values are: `true` `false` The default value is `false`.
`units`	No	The units of measure associated with the field.
`javaFormat`	No	The Java format to use when the field value is parsed. This is the Java output format used by the `NumberFormat` or `DateFormat` class.

Parent Element

Metadata, ParentDef

Child Elements

None

Example

```
<FieldDef name="Molstructure"
    type="Structure"
    molFormat="Chime"/>
<FieldDef name="Reaction Vessel ID"
    type="FixedText"
    maxLength="15" />
```

See Also

"Alphabetic List of Elements" on page 57, "Hierarchy of Elements" on page 55

Data

Contains a collection of <u>Record</u> child elements that contain the actual data. Each <u>Record</u> element contains a collection of <u>Parent</u> and <u>Field</u> elements whose hierarchical structure must match the hierarchy in the <u>Metadata</u> element. The Data element can be empty if there is no data or only the metadata is to be transferred.

Attributes

The data type of all attributes is CDATA.

Attribute	Required	Description
totalRecords	No	The total number of root-level records in the data

Parent Element

<u>Dataset</u>

Child Elements

Element	Required	Description
<u>Record</u>	No	A record of data

Example

```
<Data totalRecords="2">
   <Record>
      <Field name="Molstructure"  xml:space="preserve">
        <![CDATA[7YALEn$Akl$QPbas35quasdfsdf38...]]></Field>
      <Field name="Reaction Vessel ID">SAR6077R-01A02</Field>
      <Field name="Library ID">SAR6077R</Field>
      <Field name="Tag Id">yes</Field>
      <Parent name="XYZ Test Results" totalRecords="2">
         <Record>
            <Field name="Dose">1.0</Field>
            <Field name="Response">99.0</Field>
         </Record>
         <Record>
            <Field name="Dose">2.0</Field>
            <Field name="Response">999.0</Field>
         </Record>
      </Parent>
   </Record>
   <Record>
      <Field name="Molstructure"  xml:space="preserve">
        <![CDATA[7YALEn$Akl$QPbas35quasdfsdf38...]]></Field>
      <Field name="Reaction Vessel ID">SAR6077R-01A03</Field>
      <Field name="Library ID">SAR6077R</Field>
      <Field name="Tag Id">no</Field>
   </Record>
</Data>
```

See Also

Parent

Contains child records of data. A `Parent` can contain other `Parent` elements, thus, creating a sub-hierarchy of data.

Attributes

The data type of all attributes is `CDATA`.

Attribute	Required	Description
name	Yes	The name of the field.
totalRecords	No	The total number of records under this parent

Parent Element

Record

Child Elements

Element	Required	Description
Record	No	A record of data

Example

```
<Parent name="XYZ Test Results" ID="100">
   <Record>
      <Field name="Dose">1.0</Field>
      <Field name="Response">99.0</Field>
   </Record>
   <Record>
      <Field name="Dose">2.0</Field>
      <Field name="Response">999.0</Field>
   </Record>
</Parent>
```

See Also

Record

A record of data, and contains a set of Field elements.

Attributes

None

Parent Element

Data

Parent

Child Elements

Element	Required	Description
Field	No	A single field of data
Parent	No	Contains subrecords of data. A parent within a record creates a hierarchy.

Example

```
<Record>
   <Field name="Dose">1.0</Field>
   <Field name="Response">99.0</Field>
</Record>
```

See Also

"Alphabetic List of Elements" on page 57, "Hierarchy of Elements" on page 55

Field

Contains data for a single field in a record. Note that the `molFormat`, `molVersion`, and `rxnFormat` attributes can be used on individual `Field` elements to override the format specified in the [Metadata](#) element.

Attributes

The data type of all attributes is CDATA.

Attribute	Required	Description
name	Yes	The name of the parent field.

molFormat	No	The format for the structure, if the field contains structures. This value overrides the `molFormat` specified in the [Metadata](#) element. The valid values are:
		
		`Molfile`
		
		`Chime`
		
		`Smiles`
		
		The default value is `Molfile`.

rxnFormat	No	The format for the reaction, if the field contains reactions. This value overrides the `rxnFormat` specified in the [Metadata](#) element. The valid values are:
		
		`Rxnfile`
		
		`Chime`
		
		The default value is `Rxnfile`.

molVersion	No	The version of the data format. `molVersion` is currently only applicable to structures. The absence of this attribute implies a pre-V2000 molfile. This value overrides the `molVersion` specified in the [Metadata](#) element. The valid values are:
		
		`V2000`
		
		`V3000`

Parent Element

[Record](#)

Example

```
<Field name="Dose">1.0</Field>
```

See Also

Chapter 10: The Extended Connection Table (V3000)

Overview

The extended (V3000) molfile consists of a regular molfile "no structure" followed by a single molfile appendix that contains the body of the connection table (Ctab). The following figure shows both an alanine structure and the extended molfile corresponding to it. See Chapter 2 for the V2000 version of this same structure.

Figure 16   Extended molfile organization illustrated using alanine


```
L-Alanine
GSMACCS-II07189510252D 1   0.00366    0.00000    0        Header Block
Figure 1, J. Chem. Inf. Comput. Sci., Vol 32, No. 3., 1992  ← Comment Line
  0  0  0     0  0               999 V3000
M  V30 BEGIN CTAB
M  V30 COUNTS 6 5 0 0 1                                    Counts Line
M  V30 BEGIN ATOM
M  V30 1 C -0.6622 0.5342 0 0 CFG=2
M  V30 2 C 0.6622 -0.3 0 0
M  V30 3 C -0.7207 2.0817 0 0 MASS=13                      Atom Block
M  V30 4 N -1.8622 -0.3695 0 0 CHG=1
M  V30 5 O 0.622 -1.8037 0 0
M  V30 6 O 1.9464 0.4244 0 0 CHG=-1
M  V30 END ATOM
M  V30 BEGIN BOND
M  V30 1 1 1 2
M  V30 2 1 1 3 CFG=1
M  V30 3 1 1 4                                             Bond Block
M  V30 4 2 2 5
M  V30 5 1 2 6        Blocks not used in this Ctab:
M  V30 END BOND      Sgroup block, Rgroup block, 3D block
M  V30 END CTAB
M  END
```

Note that the "no structure" is flagged with the "V3000" instead of the "V2000" version stamp.

There are two other changes to the header in addition to the version:

- The number of appendix lines is always written as 999, regardless of how many there actually are. (All current readers will disregard the count and stop at M   END.)

- The "dimensional code" is maintained more explicitly. Thus "3D" really means 3D, although "2D" will be interpreted as 3D if any non-zero Z-coordinates are found.

Unlike the V2000 molfile, the V3000 extended Rgroup molfile has the same header format as a non-Rgroup molfile.

Note: Do **not** create a molfile with a pre-V3000 Rgroup header (`"$MDL"`, and so forth) but with V3000 Ctab blocks. This is not allowed. A pre-V3000 Rgroup molfile can only have embedded molfiles that are also pre-V3000 versions, for example, the version is either `"V2000"` or `"    "`.

Specifications For Atom and Bond Descriptions

The general syntax of an entry is:

```
M  V30 key posval posval ... [keyword=value] [keyword=value] ...
```

or

```
M  V30 BEGIN key [blockname]
M  V30 posval posval ... keyword=value keyword=value ...
...
M  V30 END key
```

Each line must begin with `"M  V30 "` with the two blank spaces after `M` and one blank space after `30`. Following this is a list of zero or more required positional values (posval). Optional values may follow which use a `'KEYWORD=value'` format. Items are separated by white space. There can also be white space preceding the first item. Trailing white space is ignored.

The value of a keyword can be a list containing two or more values:

```
KEYWORD=(N val1 val2 ... valN)
```

where `N` specifies the number of items that follow.

Values (`posval`, `value`, or `val1`, and so forth) can be strings. Strings that contain blank spaces or start with left parenthesis or double quote, must be surrounded by double quotes. A double quote may be entered literally by doubling it.

Each entry is one line of no more than 80 characters. To allow continuation when the 80-character line is too short, use a dash (–) as the last character. When read, the line is concatenated with the next line by removing the dash and stripping the initial `"M  V30"` from the following line. For example:

```
M  V30 10 20 30 "abc-
M  V30  def"
```

is read as:

```
M  V30 10 20 30 "abc  def"
```

Generally, each section of the molfile is enclosed in a *block* that consists of lines such as:

```
M  V30 BEGIN key [blockname]
...
M  V30 END key
```

The 'key' value defines the kind of block, for example, CTAB, ATOM, or BOND. Depending upon the type of block, there may or may not be values on the BEGIN line.

Conventions

The new format conventions used in this chapter are as follows:

UPPERCASE	Literal text, to be entered as shown. Only the position of "M V30 " is significant; white space may be added anywhere else to improve readability. Note that both lower- and uppercase characters, or any combination of them, are acceptable for literals. They are shown here in uppercase only for readability.
lowercase	A token, which is defined elsewhere.
[ ]	An optional item. Do not include the brackets.
[ ]*	An optional item, where there may be zero, one, two, or more of the item.
I	Separates two or more options, only one of which is valid.
/	Separates two or more items. Either or both may appear in any order.
{}	Braces are used for grouping. They indicate indefinite or definite repeat.

The Extended Connection Table

The features of the extended connection table are described in this section.

CTAB Block

A Ctab block defines the basic connection table, which is defined as:

```
M  V30 BEGIN CTAB [ctabname]
counts-line
atom-block
[bond-block]
[sgroup-block]
[3d-block]
[link-line]*
M  V30 END CTAB
```

The atom block, like the counts line, is required. The Sgroup block, 3D block, and link lines may occur in any order after the atom and bond blocks. The counts line, atom block, and bond block must appear in the order indicated.

Counts Line

A counts line is required, and must be first. It specifies the number of atoms, bonds, 3D objects, and Sgroups. It also specifies whether or not the CHIRAL flag is set. Optionally, the counts line can specify molregno. This is only used when the regno exceeds 999999 (the limit of the format in the molfile header line). The format of the counts line is:

```
M  V30 COUNTS na nb nsg n3d chiral [REGNO=regno]
```

where:

na	= number of atoms
nb	= number of bonds
nsg	= number of Sgroups
n3d	= number of 3D constraints
chiral	= 1 if molecule is chiral, 0 if not
regno	= molecule or model regno

Atom Block

An atom block specifies all node information for the connection table. It must precede the bond block. It has the following format:

```
M  V30 BEGIN ATOM
M  V30 index type x y z aamap -
M  V30 [CHG=val] [RAD=val] [CFG=val] [MASS=val] -
M  V30 [VAL=val] -
M  V30 [HCOUNT=val] [STBOX=val] [INVRET=val] [EXACHG=val] -
M  V30 [SUBST=val] [UNSAT=val] [RBCNT=val] -
M  V30 [ATTCHPT=val] -
M  V30 [RGROUPS=(nvals val [val ...])] -
M  V30 [ATTCHORD=(nvals nbr1 val1 [nbr2 val2 ...])] -
. . .
M  V30 END ATOM
```

The values are described in the following table.

Figure 17   Meaning of values in the atom block

Field	Meaning	Values	Notes
index	Atom index	Integer > 0	Identifies atoms. The actual value of the index does not matter as long as each index is unique to each atom. However, extremely large numbers used as indexes can cause the program to fail to allocate memory for the correspondence array.
type	Atom type	Type = reserved atom *or* atom *or* [NOT] '['atom, atom,...']' where reserved atom = R# = Rgroup A = "any" atom Q = any atom but C or H * = "star" atom Atom = character string (For example, 'C' or 'Cl')	A character string. If the string contains white space, it must be quoted. It can be a single atom or an atom list enclosed in square brackets with an optional preceding NOT.
x y z	Atom coordinates	Angstroms	

aamap	Atom-atom mapping	0 = no mapping > 0 = mapped atom	Reaction property
CHG	Atom charge	Integer 0 = none (default)	Same range as V2000
RAD	Atom radical	0 = none (default) 1 = singlet 2 = doublet 3 = triplet	
CFG	Stereo configuration	0 = none (default) 1 = odd parity 2 = even parity 3 = either parity	
MASS	Atomic weight		Default = natural abundance A specified value indicates the absolute atomic weight of the designated atom.
VAL	Valence	Integer > 0 *or* 0 = none (default) -1 = zero	Abnormal valence
HCOUNT	Query hydrogen count	Integer > 0 *or* 0 = not specified (default) -1 = zero	Same maximum value as V2000
STBOX	Stereo box	0 = ignore the configuration of this double bond atom (default) 1 = consider the stereo configuration of this double bond atom	Both atoms of a double bond must be marked to search double bond stereochemistry. Alternatively, the STBOX bond property can be used.
INVRET	Configuration inversion	0 = none (default) 1 = configuration inverts 2 = configuration retained	Reaction property
EXACHG	Exact change	0 = property not applied (default) 1 = exact change as displayed in the reaction	Reaction property
SUBST	Query substitution count	Integer > 0 *or* 0 = not specified (default) -1 = none	Same maximum value as V2000.
UNSAT	Query unsaturation flag	0 = not specified (default) 1 = unsaturated	
RBCNT	Query ring bond count	Integer > 0 *or* 0 = not specified (default) -1 = none	Same maximum value as V2000.

ATTCHPT	Rgroup member attachment points	Attachment points on member: -1 = first and second site 1 = first site only 2 = second site only	When the Rgroup member atom has two attachment points, the atom with the lowest index number attaches to the first attachment point
RGROUPS	nvals is the number of Rgroups that comprise this R# atom. val is the Rgroup number.	Integer > 0 Integer > 0	
ATTCHORD	nvals is the number of values that follow on the ATTCHORD line nbr1 is atom neighbor index #1, nbr2 is index #2 val1 is the attachment order for the nbr1 attachment.	Integer > 0	A list of atom neighbor index and atom neighbor value pairs that identify the attachment order information at the R# atom

Bond block

A bond block specifies all edge information for the connection table. It must precede the Sgroup or 3D blocks. Its format is:

```
M  V30 BEGIN BOND
M  V30 index type atom1 atom2 [CFG=val] [TOPO=val] [RXCTR=val]
[STBOX=val]
...
M  V30 END BOND
```

where the values are described in the following table:

Figure 18   Meaning of values in the bond block

Field	Meaning	Values	Notes
index	Bond index	Integer > 0	The actual value of the index does not matter as long as all are unique. However, extremely large numbers used as indexes can cause the program to fail to allocate memory for the correspondence array.
type	Bond type	Integer: 1 = single 2 = double 3 = triple 4 = aromatic 5 = single or double 6 = single or aromatic 7 = double or aromatic 8 = any	Types 4 through 8 are for queries only.

atom1,atom2	Atom indexes	Integer > 0	Atom1 and Atom2 are bond end points.
CFG	Bond configuration	0 = none (default) 1 = up 2 = either 3 = down	
TOPO	Query property	0 = not specified (default) 1 = ring 2 = chain	
RXCTR	Reacting center status	0 = unmarked (default) -1 = not a reacting center 1 = generic reacting center Additional: 2 = no change 4 = bond made or broken 8 = bond order changes 12 =(4 + 8) bond made or broken and changes 5 = (4 + 1), 9 = (8 + 1), and 13 =(12 + 1) are also possible	
STBOX	Stereo box	0 = ignore the configuration of this double bond (default) 1 = consider the stereo configuration of this double bond	A double bond must be marked to search double bond stereochemistry

Link atom line

There is one link atom line for each link atom in the Ctab. A link atom line has the format:

```
M  V30 LINKNODE minrep maxrep nbonds inatom outatom [inatom
outatom...]
```

Figure 19 Meaning of values in link lines

Field	Meaning	Values	Notes
minrep	Minimum number of group repetitions.	1	For future expansion. Not currently used.
maxrep	Maximum number of group repetitions.	Integer > 0	
nbonds	Number of directed bonds defining the group.	nbonds = # of pairs of inatom-outatom tuples	Number of tuples is usually two but may be one for link nodes with an attachment point.
inatom	Atom index of atom in the repeating group.	Integer > 0	
outatom	Atom index of atom bonded to inatom, but outside of repeating group.	Integer > 0	

Sgroup block

The Sgroup block contains general Sgroup information and information on each Sgroup structure as shown here. For the V2000 version of this Sgroup structure and connection table, see Chapter 2.

Figure 20   Connection Table Organization of an Sgroup Structure
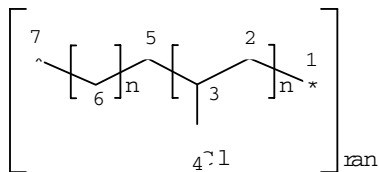

```
Polymer
GSMACCS-II07189510252D 1   0.00374    0.00000    0                    ─── Header Block
Figure 5, J. Chem. Inf. Comput. Sci., Vol 32, No. 3., 1992   ←──── Comment Line
  0  0  0    0  0              999 V3000
M  V30 BEGIN CTAB
M  V30 COUNTS 7 6 3 0 0                                               ─── Counts Line
M  V30 BEGIN ATOM
M  V30 1 * 2.9463 0.3489 0 0
M  V30 2 C 1.6126 1.1189 0 0
M  V30 3 C 0.2789 0.3489 0 0 CFG=3        Blocks not used in this Ctab:
M  V30 4 Cl 0.2789 -1.1911 0 0            Rgroup block, 3D block        ─── Atom Block
M  V30 5 C -1.0548 1.119 0 0
M  V30 6 C -2.3885 0.349 0 0
M  V30 7 * -3.9246 1.147 0 0
M  V30 END ATOM
M  V30 BEGIN BOND
M  V30 1 1 1 2
M  V30 2 1 2 3
M  V30 3 1 3 4
M  V30 4 1 5 6                                                        ─── Bond Block
M  V30 5 1 5 3
M  V30 6 1 7 6
M  V30 END BOND
M  V30 BEGIN SGROUP                                                   ─── Sgroup Block
M  V30 1 SRU 5 ATOMS=(2 5 6) XBONDS=(2 5 6) BRKXYZ=(9 -0.6103 1.2969 0 -0.6103 -   ─── Sgroup 1
M  V30 0.171 0 0 0 0) BRKXYZ=(9 -3.1565 0.185 0 -3.1565 1.311 0 0 0 0) -   ←──── Properties
M  V30 CONNECT=HT
M  V30 2 SRU 6 ATOMS=(3 2 3 4) XBONDS=(2 1 5) BRKXYZ=(9 2.2794 1.2969 0 2.2794 -   ─── Sgroup 2
M  V30 0.1709 0 0 0 0) BRKXYZ=(9 -0.1657 0.171 0 -0.1657 1.2969 0 0 0 0) -   ←──── Properties
M  V30 CONNECT=HT
M  V30 3 COP 7 ATOMS=(7 1 2 3 4 5 6 7) BRKXYZ=(9 3.6382 1.6391 0 3.6382 -   ─── Sgroup 3
M  V30 -1.7685 0 0 0 0) BRKXYZ=(9 -4.707 -1.7685 0 -4.707 1.6391 0 0 0 0) -   ←──── Properties
M  V30 SUBTYPE=RAN
M  V30 END SGROUP
M  V30 END CTAB
M  END
```

An Sgroup block defines all Sgroups in the molecule, including superatoms. The format is as follows:

```
M  V30 BEGIN SGROUP
[M  V30 DEFAULT [CLASS=class] -]
M  V30 index type extindex -
M  V30 [ATOMS=(natoms atom [atom ...])] -
M  V30 [XBONDS=(nxbonds xbond [xbond ...])] -
M  V30 [CBONDS=(ncbonds cbond [cbond ...])] -
M  V30 [PATOMS=(npatoms patom [patom ...])] -
M  V30 [SUBTYPE=subtype] [MULT=mult] -
M  V30 [CONNECT=connect] [PARENT=parent] [COMPNO=compno] -
M  V30 [XBHEAD=(nxbonds xbond [xbond ...])] -
M  V30 [XBCORR=(nxbpairs xb1 xb2 [xb1 xb2 ...])] -
M  V30 [LABEL=label] -
M  V30 [BRKXYZ=(9 bx1 by1 bz1 bx2 by2 bz2 bx3 by3 bz3])* -
M  V30 [ESTATE=estate] [CSTATE=(4 xbond cbvx cbvy cbvz)]* -
M  V30 [FIELDNAME=fieldname] [FIELDINFO=fieldinfo] -
M  V30 [FIELDDISP=fielddisp] -
M  V30 [QUERYTYPE=querytype] [QUERYOP=queryop] -
M  V30 [FIELDDATA=fielddata] ... -
M  V30 [CLASS=class] -
M  V30 [SAP=(3 aidx lvidx id)]* -
M  V30 [BRKTYP=bracketType] -
...
M  V30 END SGROUP
```

The DEFAULT field provides a way to specify default values for keyword options. The same keyword options and values as defined in the following table.

Figure 21 Meaning of values in the Sgroup block

Field	Meaning	Values	Notes
index	Sgroup index	Integer > 0	The actual value of the index does not matter as long as all indexes are unique. However, extremely large numbers used as indexes can cause the program to fail to allocate memory for the correspondence array.
type	Sgroup type	String. Only first 3 letters are significant: SUPeratom MULtiple SRU MONomer COPolymer CROsslink MODification GRAft COMponent MIXture FORmulation DATa ANY GENeric	
extindex	External index value	Integer => 0: If 0, positive integer assigned	Use 0 to autogenerate a number. This is the V2000 Sgroup label

ATOMS	natoms is the number of atoms that define the Sgroup.	Integer > 0	
	atom is the atom index.	Integer > 0	
XBONDS	nxbonds is the number of crossing bonds.	Integer > 0	
	xbond is the crossing-bond index.	Integer > 0	
CBONDS	ncbonds is the number of containment bonds.	Integer > 0	Only used for Data Sgroups.
	cbond is the containment-bond index	Integer > 0	
PATOMS	npatom is the number of paradigmatic repeating unit atoms.	Integer > 0	This field is expected to become obsolete and is retained for compatibility with MACCS-II. The field is only used for multiple groups.
	patom is the atom index of an atom in the paradigmatic repeating unit for a multiple group.		
SUBTYPE	subtype is the Sgroup subtype.	String. Only the first 3 letters are significant: ALTernate RANdom BLOck	
MULT	mult is the multiple group multiplier.	Integer > 0	
CONNECT	connect is the connectivity.	String values are as follows: EU (default) HH HT	The default, if missing, is EU. The MDL V2000 writer never writes an EU entry.
PARENT	parent is the parent Sgroup index.	Integer > 0	
COMPNO	compno is the component order number.	Integer > 0	
XBHEAD	nxbonds is the number of crossing bonds that cross the "head" bracket.	Integer > 0	
	xbond is the crossing-bond index.	Integer > 0	If XBHEAD is missing, no bonds are paired as the head or tail of the repeating unit

XBCORR	nxbpairs	2 x the number of pairs of crossing-bond correspondence, that is, the number of values in list.	
	xb1 - xb2 is the pairs of crossing-bond correspondence, that is, xb1 connects to xb2.	Integer > 0	
LABEL	label is the display label for this Sgroup.	String	For example, superatom name
BRKXYZ	bx1 - bz3 are the double (X,Y,Z) display coordinates in each bracket.	Angstroms	By specifying 3 triples, the format allows a 3D display. However, only the first two (X, Y) coordinates are currently used. The Z value and last (X, Y) coordinates are currently ignored and should be set to zero.
ESTATE	estate is the expanded display state information for superatoms.	String E = expanded superatom or multiple group	Only superatoms and multiple groups (shortcuts) in an expanded internal state are supported. This field defines whether a superatom or multiple group is displayed as expanded or contracted. This field is expected to become obsolete.
CSTATE	xbond is the crossing bond of the expanded superatom.	Integer > 0	Display vector information for the contracted superatom.
	cbvx - cvbz is the vector to contracted superatom.	Angstroms	Only present for expanded superatoms. One CSTATE entry per crossing bond.
FIELDNAME	fieldname is the name of data field for Data Sgroup.	String	
FIELDINFO	fieldinfo is the program-specific field information.	Free-format string	Example: In MACCS-II this is: "\<type\> \<units/format\>"
FIELDDISP	fielddisp is the Data Sgroup field display information.	Free-format string	This string is interpreted by V3000 as identical to V2000 appendix for Data Sgroup display ('M SDD') except for the index value.
QUERYTYPE	querytype is the type of query or no query if missing.	String ' ' = not a query (default) 'MQ' = MACCS-II query 'IQ' = ISIS query \<p\>Q' = \<program\> query	
QUERYOP	queryop is the query operator.	String. ISIS: query operator MACCS-II: blank or missing	Example: "=" or "LIKE" in ISIS
FIELDDATA	fielddata is the query or field data.	Free-format string	Only one entry per query, but can be more than one for actual data. The order of the entries is important.
CLASS	class is the character string for superatom class.	String	Example: PEPTIDE

SAP	aidx is the index of attachment point or potential attachment point atom.	Integer > 0	
	lvidx is the index of leaving atom.	Allowed integers are: 0 = none or implied H 'aidx' = atom index number of attachment point atom > 0 = atom index number of atom bonded to 'aidx'	
	id is the attachment identifier.	String (two chars in V2000)	There must be multiple entries if superatom has more than one attachment point. The order of the entries defines the order of the attachment points. Note that SAP entries may or may not include the actual attachment points, depending on the particular superatom and its representation on MDL ISIS/Desktop.
BRKTYP	bracketType is the displayed bracket style.	Allowed values for this string are: BRACKET (default) PAREN	This information supports Sgroup enhancements on MDL ISIS/Desktop.

Correspondence with existing V2000 appendices

```
M  STY = type
M  SST = SUBTYPE
M  SLB = extindex
M  SCN = CONNECT
M  SDS = ESTATE
M  SAL = ATOMS
M  SBL = XBONDS or CBONDS
M  SPA = PATOMS
M  SMT = LABEL and MULT
M  CRS = XBHEAD,XBCORR
M  SDI = BRKXYZ
M  SBV = CSTATE
M  SDT = FIELDNAME, FIELDINFO, QUERYTYPE, QUERYOP
M  SDD = FIELDDISP
M  SCD = (not required)
M  SED = FIELDDATA
M  SPL = PARENT
M  SNC = COMPNO
M  SAP = SAP
M  SCL = CLASS
M  SBT = BRKTYP
```

Collection block

A collection block specifies all collection information for objects in the current connection table context. Collection blocks must be provided after the blocks that define the objects included in the collection to minimize the amount of forward object references that must be maintained by the file reader.

```
M  V30 BEGIN COLLECTION
[M  V30 DEFAULT -]M  V30 name/subname -
M  V30 [ATOMS=(natoms atom [atom ...])] -
M  V30 [BONDS=(nbonds bond [bond ...])] -
M  V30 [SGROUPS=(nsgroups sgrp [sgrp ...])] -
M  V30 [OBJ3DS=(nobj3ds obj3d [obj3d ...])] -
M  V30 [MEMBERS=(nmembers member [member ...])] -
M  V30 [RGROUPS=(nrgroups rgroup [rgroup ...])] -
...
M  V30 END COLLECTION
```

Figure 22   Meaning of values in the collection block

Field	Meaning	Value	Notes
name/subname	Collection id	Nonblank string	
ATOMS	natoms is the number of atoms included in the collection	Integer > 0	
	atom is the atom index	Integer > 0	
BONDS	nbonds is the number of bonds included in the collection	Integer > 0	
	bond is the bond index	Integer > 0	
SGROUPS	nsgroups is the number of sgroups included in the collection	Integer > 0	
	sgrp is the sgroup index	Integer > 0	
OBJ3DS	nobj3ds is the number of 3D features included in the collection	Integer > 0	
	obj3d is the 3D feature index	Integer > 0	
MEMBERS	nmembers is the number of members included in the collection	Integer > 0	
	member is the member identifier	ROOT or RrMm	r > 0, m > 0
RGROUPS	nrgroups is the number of rgroups included in the collection	Integer > 0	
	rgroup is the rgroup identifier	Rr	r > 0

- Collections can be named in a semi-arbitrary fashion.

- A two part name is supported for collections on import and export.

- The subname designation is required.

- The name is the unique identifier for the collection contents.

- All collections of the same name are presumed to indicate various pieces of the same collection, which may be provided in one or more COLLECTION block entries provided within various subblocks of the full connection table.

- Collection names are **not** case sensitive, **must** contain only printable characters, and **must** begin with an alphanumeric character.

- Collection objects are presumed to be unordered, and no preservation of input order is necessary or required on subsequent file writes.

- Future enhancements to the collection block information will provide ordered collection support.

The default delimiter for the collection name is the '/' character. If the name begins with a non-alphanumeric character, it is presumed to indicate an alternate delimiter character, thus a collection name that contains a '/' character might be designed as '|name|/subname/|', for example. In addition, if the first character is '"', it is presumed that the collection name is being quoted due to the presence of spaces.

The name MDLV30 is a reserved name used to designate internal collections. User specified collections may use any other arbitrary naming conventions. The default action for all MDL provided V3000 readers and writers is to preserve all user collection information for CTfile import/export operations. Internal collections may also be preserved if their contents have been validated as correct input for the specified internal representation. There is no implied validation for user collections other than requiring the collection to refer to valid objects. Collections may not contain other collections in their definition.

The default action on registration is to strip user collection information without error, but possibly with status or warning messages being issued. The internal collection types are:

MDLV30/HILITE	Highlighting collection. Default action for renderers is to provide a "highlighted" display of the specified objects. All object types are allowed in this collection: atoms, bonds, Sgroups, 3Dfeatures, Rgroups, members, and components.
MDLV30/STEABS	Absolute stereochemistry collection. This collection defines the set of stereocenters in the structure that have absolute configurations. This is an atom collection.
MDLV30/STERACn	"Racemic" stereochemistry collection ($n>0$). This collection defines a set of stereogenic centers whose relative configuration is known. A mixture of the two enantiomeric relative configurations is present. This is an atom collection.
MDLV30/STERELn	Relative stereochemistry collection ($n>0$). This collection defines a set of stereogenic centers whose relative configuration is known. Only one of the two enantiomeric relative configurations is present. There is no assumption about which of the two configurations is present. This is an atom collection.

3D block

The 3D block contains 3D information as shown below. For the V2000 version of this 3D query and its connection table, see Chapter 2.3D Block
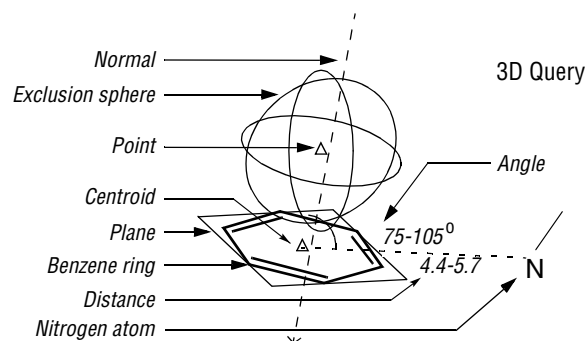

```
3D Query                                                      ──────── Header Block
  MACCS-II07189510253D 1   1.00000     0.00000     0
Figure 6, J. Chem. Inf. Comput. Sci., Vol 32, No. 3., 1992  ← ─── Comment Line
   0  0  0     0  0              999 V3000
M  V30 BEGIN CTAB
M  V30 COUNTS 8 7 0 7 0                                       ──────── Counts Line
M  V30 BEGIN ATOM
M  V30 1 C 1.0252 0.2892 1.1122 0
M  V30 2 C -0.4562 0.6578 1.3156 0
M  V30 3 C -1.4813 0.3687 0.2033 0
M  V30 4 C -1.0252 -0.2892 -1.1122 0
M  V30 5 C 0.4562 -0.6578 -1.3156 0                           ──────── Atom Block
M  V30 6 C 1.4813 -0.3687 -0.2033 0
M  V30 7 N 4.1401 -0.1989 1.3456 0
M  V30 8 C 4.6453 0.5081 1.7417 0
M  V30 END ATOM
M  V30 BEGIN BOND
M  V30 1 1 1 2
M  V30 2 2 2 3
M  V30 3 1 3 4
M  V30 4 2 4 5       Blocks not used in this Ctab: Sgroup block,
M  V30 5 1 5 6       Rgroup block                              ──────── Bond Block
M  V30 6 2 6 1
M  V30 7 1 7 8
M  V30 END BOND
M  V30 BEGIN OBJ3D
M  V30 1 -7 6 "" 0 0 BASIS=(3 6 4 2)
M  V30 2 -5 13 "" 0 0 BASIS=(6 1 2 3 4 5 6)
M  V30 3 -8 7 "" 0 0 BASIS=(2 O3D.1 O3D.2)
M  V30 4 -3 6 "" -2 0 BASIS=(2 O3D.1 O3D.3) PNTDIR=1          ──────── 3D object block
M  V30 5 -16 12 "" 1.5 0 BASIS=(1 O3D.4) UNCONNOK=1
M  V30 6 -12 10 "" 75 105 BASIS=(3 O3D.4 O3D.1 7)
M  V30 7 -9 3 "" 4.4 5.7 BASIS=(2 7 O3D.1)
M  V30 END OBJ3D
M  V30 END CTAB
M  END
```
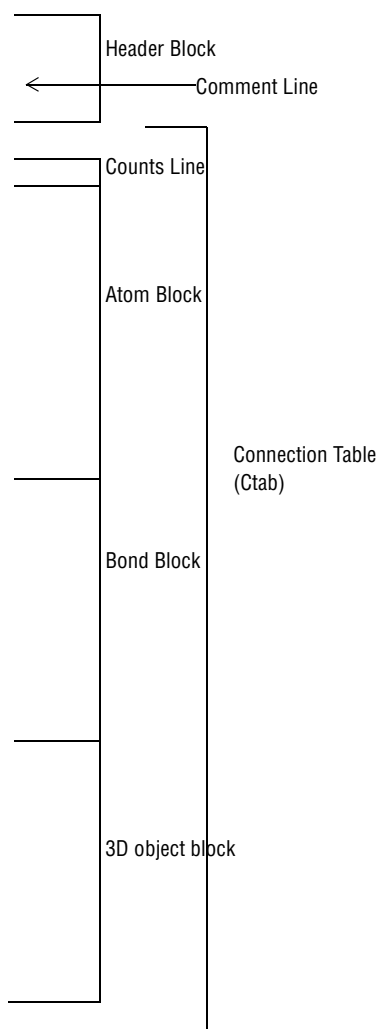
A 3D block specifies information for all 3D objects in the connection table. It must follow the atom and bond blocks. As in V2000 molfiles, there can be only one fixed-atom constraint.

The format of the 3D block is as follows:

```
M  V30 BEGIN OBJ3D
M  V30 index type color name value1 value2 -
M  V30 BASIS=(nbvals bval [bval ...]) -
M  V30 [ALLOW=(nvals val [val ...])] [PNTDIR=val] [ANGDIR=val] -
M  V30 [UNCONNOK=val] [DATA=strval] -
M  V30 [COMMENT=comment]
...
M  V30 END OBJ3D
```

Figure 23   Meaning of values in the 3D block

Field	Meaning	Values	Notes
index	3D object index	Integer > 0	The actual value of the index does not matter as long as all indexes are unique. However, extremely large numbers used as indexes can cause the program to fail to allocate memory for the correspondence array.
type	Object type	Integer < 0 for geometric constraints and for data constraints Integer > 0 are field IDs	This format is the same as V2000.
color	Color value	Integer > 0	
name	Object name or, for data query, the field name.	String	
value1	Distance, radius, deviation, or minimum value.	Floating point, value1 = 0 if constraint has no floating values	
value2	Maximum value for range constraints.	Floating point, value2 = 0 if not a range constraint	
BASIS	nbvals is the number of objects in basis.	Integer > 0	
	bval is the atom number or 3D object index	Integer or O3D.integer	For objects where order is important, for example, in an angle constructed from three points, the order must be the same as in V2000 molfiles.
ALLOW	nvals is the number of atoms allowed in an exclusion sphere.	Integer > 0	
	val is the atom number.	Integer > 0	
PNTDIR		0 = point has no direction 1 = point has direction	
ANGDIR		0 = dihedral angle has no direction 1 = dihedral angle has direction	MACCS-II uses 'Chiral'.
UNCONNOK		0 = unconnected atoms are not OK 1 = unconnected atoms are OK	
DATA	strval is the data query string	String	
COMMENT	string comment	String. Normally uses the MACCS-II DASP, DISP, and BOX values	Same as V2000 molfile

The Extended Rgroup Query Molfile

A single molecule or Rgroup molecule connection table. The header is contained in the normal header location, that is, in the first three lines of the file. The body of the new molecule is contained in new appendixes, organized as follows:A molecule block consists of a main Ctab, plus optionally one or more Rgroup definitions.

```
ctab-block
[rgroup-block]*
```

Rgroup block

The Rgroup file shown in the next figure corresponds to the following Rgroup query. For the V2000 version of the Rgroup query and its connection table, see Chapter 5

Figure 24   Connection table organization of an Rgroup query.
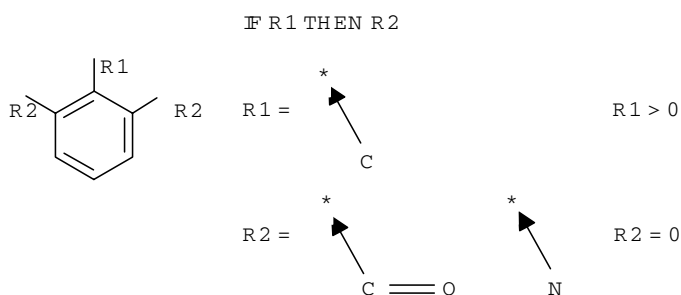
```
GSMACCS-II07189510252D 1   0.00353    0.00000    0                    Header Block

  0  0  0    0  0              999 V3000
M  V30 BEGIN CTAB
M  V30 COUNTS 9 9 0 0 0                                                Counts Line
M  V30 BEGIN ATOM
M  V30 1 C 1.3337 0.77 0 0
M  V30 2 C 0 1.54 0 0
M  V30 3 C -1.3337 0.77 0 0
M  V30 4 C -1.3337 -0.77 0 0                                          Atom Block
M  V30 5 C 0 -1.54 0 0                                                for root
M  V30 6 C 1.3337 -0.77 0 0
M  V30 7 R# 0 3.08 0 0 RGROUPS=(1 1)
M  V30 8 R# 2.6674 1.54 0 0 RGROUPS=(1 2)
M  V30 9 R# -2.6674 1.54 0 0 RGROUPS=(1 2)
M  V30 END ATOM
M  V30 BEGIN BOND
M  V30 1 1 1 2
M  V30 2 2 2 3
M  V30 3 1 3 4
M  V30 4 2 4 5
M  V30 5 1 5 6                                                        Bond Block
M  V30 6 2 6 1                                                        for root
M  V30 7 1 1 8
M  V30 8 1 2 7
M  V30 9 1 3 9
M  V30 END BOND
M  V30 END CTAB
M  V30 BEGIN RGROUP 1
M  V30 RLOGIC 2 0 ""                                                  Rgroup logic Line
M  V30 BEGIN CTAB
M  V30 COUNTS 1 0 0 0 0                          Counts Line
M  V30 BEGIN ATOM
M  V30 1 C 12.21 14.3903 0 0 ATTCHPT=1          Atom Block
M  V30 END ATOM
M  V30 END CTAB
M  V30 END RGROUP
M  V30 BEGIN RGROUP 2
M  V30 RLOGIC 0 0 0                                                   Rgroup logic Line
M  V30 BEGIN CTAB
M  V30 COUNTS 2 1 0 0 0                          Counts Line
M  V30 BEGIN ATOM
M  V30 1 C -1.4969 0.0508 0 0 ATTCHPT=1         Atom Block
M  V30 2 O 0.0431 0.0508 0 0
M  V30 END ATOM
M  V30 BEGIN BOND
M  V30 1 2 1 2                                   Bond Block
M  V30 END BOND
M  V30 END CTAB
M  V30 BEGIN CTAB
M  V30 COUNTS 1 0 0 0 0                          Counts Line
M  V30 BEGIN ATOM
M  V30 1 N 12.21 14.3903 0 0 ATTCHPT=1          Atom Block
M  V30 END ATOM
M  V30 END CTAB
M  V30 END RGROUP
M  END
```

Annotations (right side): Header Block; Counts Line; Atom Block for root; Bond Block for root; Ctab for Rgroup root; Rgroup logic Line; Ctab for Rgroup 1 Member 1; Block for Rgroup 1; Rgroup logic Line; Ctab for Rgroup 2 Member 1; Ctab for Rgroup 2 Member 2; Block for Rgroup 2

An Rgroup block defines one Rgroup. Each Ctab block specifies one member.

```
M  V30 BEGIN RGROUP rgroup-number
[rgroup-logic-line]
ctab-block
[ctab-block]*
M  V30 END RGROUP
```

Figure 25   Meaning of values in the Rgroup block

Field	Meaning	Values	Notes
rgroup-number	Index of this rgroup	Integer > 0	

Rgroup logic lines

There is zero or one Rgroup logic line for each Rgroup in the molecule. If present, the Rgroup logic line specifies if-then logic between Rgroups, the convention about unfilled valence sites, and the Rgroup occurrence information. Its format is:

```
M  V30 RLOGIC thenR RestH Occur
```

Figure 26   Meaning of values in Rgroup logic line

Field	Meaning	Values	Notes
thenR	Number of a "then" Rgroup	0 = none (default)	
RestH	Attachment(s) at Rgroup position	0 = off, that is, any molecule fragment at any unsatisfied Rgroup location (default) 1 = only hydrogen or a member of Rgroup is allowed	
Occur	String specifying number (range) of Rgroup occurrence sites that need to be satisfied.	String '> 0' = default	Similar to MACCS-II and ISIS: [N[,[N[,...]]]]

Automatic V3000 output

Current Elsevier MDL products support reading and writing of V2000 and V3000 formats. These products will preferentially write V2000 molfiles to maximize interoperability with third party applications. However, because of the limitations imposed by the V2000 format, there are situations when the V3000 format will be automatically used.

The following are cases in which the V3000 format is automatically written instead of the V2000 format:

• Structure highlighting - The V3000 format is required for molecule or collection highlighting. See the "Collection block" on page 92 which contains information about structure highlighting.

• Enhanced stereochemistry features - The V3000 format is required when using the enhanced stereochemical representations. See the "Collection block" on page 92 which contains information about the enhanced stereochemistry features.

- The maximum length of a field with a fixed width is exceeded - If any of the fields with fixed widths for any connection table properties are exceeded, then the V3000 format will be used. For example, if the number of atoms (or bonds) exceeds 999, the V3000 format will be used. This is because the number of atoms (or bonds) on the V2000 counts line cannot exceed 3 columns (see "The Counts Line" on page 12). For more details about the fixed field widths in the V2000 format, see "Chapter 2: The Connection Table [CTAB] (V2000)" on page 11.

Chapter 11: The Extended Reaction File

Overview

Rxnfiles contain structural data for the reactants and products of a reaction. For the V2000 version of this reaction file, see Chapter 7.

The first line of the file begins with $RXN to identify the file as a reaction file. The V3000 version follows the $RXN token to indicate the extended version of the reaction file. There are two new block types introduced: REACTANT and PRODUCT.

Figure 27   V3000 Rxnfile for the acylation of benzene

```
$RXN V3000                                                      ─┐ Header Block

                0503021738    7439                               │

M  V30 COUNTS 2 1                                               ─── Counts Line
M  V30 BEGIN REACTANT
M  V30 BEGIN CTAB
M  V30 COUNTS 4 3 0 0 0               ── Counts Line      ─┐
M  V30 BEGIN ATOM                                          │
M  V30 1 C 0.323 -0.2377 0 1                               │
M  V30 2 C -1.0362 -0.9618 0 2        Atom Block           │
M  V30 3 O 0.323 1.4154 0 3                                │  Ctab for
M  V30 4 Cl 1.6423 -1.0307 0 0                             │  first reactant
M  V30 END ATOM                                            │
M  V30 BEGIN BOND                                          │
M  V30 1 1 1 2 RXCTR=2                                     │
M  V30 2 2 1 3 RXCTR=2                 Bond Block           │
M  V30 3 1 1 4 RXCTR=4                                     │
M  V30 END BOND                                            │
M  V30 END CTAB                                           ─┘
M  V30 BEGIN CTAB
M  V30 COUNTS 6 6 0 0 0               ── Counts Line      ─┐
M  V30 BEGIN ATOM                                          │
M  V30 1 C 1.3331 -0.7694 0 5                              │
M  V30 2 C 1.3331 0.7694 0 6                               │
M  V30 3 C 0 -1.5417 0 7                                   │
M  V30 4 C 0 1.5417 0 8                Atom Block           │
M  V30 5 C -1.3331 -0.7694 0 9                             │
M  V30 6 C -1.3331 0.7694 0 10                             │  Ctab for
M  V30 END ATOM                                            │  second reactant
M  V30 BEGIN BOND                                          │
M  V30 1 1 1 2 RXCTR=2                                     │
M  V30 2 2 1 3 RXCTR=2                                     │
M  V30 3 2 2 4 RXCTR=2                 Bond Block           │
M  V30 4 1 3 5 RXCTR=2                                     │
M  V30 5 1 4 6 RXCTR=2                                     │
M  V30 6 2 5 6 RXCTR=2                                     │
M  V30 END BOND                                            │
M  V30 END CTAB                                           ─┘
M  V30 END REACTANT
M  V30 BEGIN PRODUCT
M  V30 BEGIN CTAB
M  V30 COUNTS 9 9 0 0 0               ── Counts Line      ─┐
M  V30 BEGIN ATOM                                          │
M  V30 1 C -0.5331 -0.1358 0 5                             │
M  V30 2 C -1.8606 0.633 0 6                               │
M  V30 3 C -0.5331 -1.6992 0 7                             │
M  V30 4 C 0.8201 0.6305 0 1           Atom Block           │
M  V30 5 C -3.2189 -0.1358 0 8                             │  Ctab for
M  V30 6 C -1.8811 -2.4731 0 9                             │  product
M  V30 7 C 2.1297 -0.1128 0 2                              │
M  V30 8 O 0.8534 2.2297 0 3                               │
M  V30 9 C -3.2292 -1.6863 0 10                            │
M  V30 END ATOM                                            │
M  V30 BEGIN BOND                                          │
M  V30 1 1 1 2 RXCTR=2                                     │
M  V30 2 2 1 3 RXCTR=2                                     │
M  V30 3 1 1 4 RXCTR=4                                     │
M  V30 4 2 2 5 RXCTR=2                                     │
M  V30 5 1 3 6 RXCTR=2                 Bond Block           │
M  V30 6 1 4 7 RXCTR=2                                     │
M  V30 7 2 4 8 RXCTR=2                                     │
M  V30 8 1 5 9 RXCTR=2                                     │
M  V30 9 2 6 9 RXCTR=2                                     │
M  V30 END BOND                                            │
M  V30 END CTAB                                           ─┘
M  V30 END PRODUCT
M  END
```

Reactant block

Product block

Appendix A: Stereo Notes

Parity is illustrated as follows:

Mark a bond attached at a stereo center Up or Down to define the configuration

Number the atoms surrounding the stereo center with 1, 2, 3, and 4 in order of increasing atom number (position in the atom block) (a hydrogen atom should be considered the highest numbered atom, in this case atom 4). View the center from a position such that the bond connecting the highest-numbered atom (4) projects behind the plane formed by atoms 1, 2, and 3.

Note: In the figure, atoms 1, 2, and 4 are all in the plane of the paper, and atom 3 is above the plane.
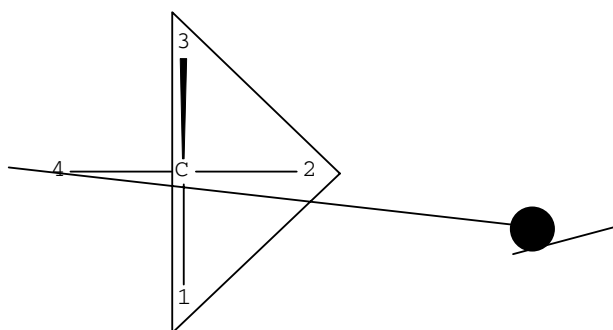
Sighting towards atom number 4 through the plane (123), you see that the three remaining atoms can be arranged in either a clockwise or counterclockwise direction in ascending numerical order.
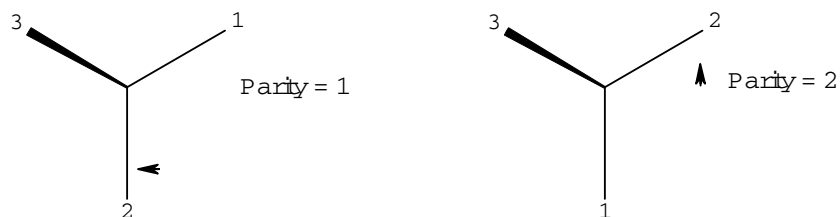
The Ctab lists a parity value of 1 for a clockwise arrangement at the stereo center and 2 for counterclockwise. A center with an Either bond has a parity value of 3. An unmarked stereo center is also assigned a value of 3. The first example above has a parity value of 2.

Index