Teradata DBMS

Quick Reference Guide

SYNTAX CONVENTIONS

- Uppercase characters indicate keywords.
- Lower case characters indicate that a value or name is to be substituted in their place.
- Underscores indicate the default value.
- Special characters, including blanks, are required as shown unless specified otherwise.
- Braces {} indicate a choice of options; one of these choices must be entered.
- A vertical bar | indicates alternatives (same as braces, but on one line)
- Brackets [] indicate an optional entry.
- Horizontal ellipses indicate a phrase that can be repeated.
- Vertical ellipses indicate omitted portions of the statement or command.

SQL statements are listed alphabetically. Defaults are underscored.

```
ABORT ['msgtext'] [FROM tname] [WHERE cond]
```

```
ALTER TABLE [dbname.]tname [,option [... ,option] ]

   [ {      {       {datadesc  }           }                                    } ]
   [ { ADD  { cname {constraint}           }                                    } ]
   [ {      {       {NULL      }           }                                    } ]
   [ {      { [CONSTRAINT name] constraint }                                    } ]

   [ { ADD RANGE BETWEEN range [..., range] [,NOT IN RANGE [OR UNKNOWN]] [,UNKNOWN]     } ]

   [ {      { cname                        }                                    } ]
   [ { DROP { [cname          ] constraint }                                    } ]
   [ {      { [CONSTRAINT name]           }                                    } ]
   [ {      { INCONSISTANT REFERENCES     }                                    } ]

   [ { DROP RANGE { WHERE expr [..., expr]     }                               } ]
                  { BETWEEN range [..., range] }[,NOT IN RANGE [OR UNKNOWN]] [,UNKNOWN]

   [ { MODIFY {cname          } CHECK expr                                      } ]
   [ {        {CONSTRAINT name}                                               } ]

   [ { MODIFY [UNIQUE] PRIMARY INDEX [idxname | (cname [..., cname])]          } ]
              { NOT PARTITIONED                                          }
              { PARTITION BY partexpr [WITH {DELETE | INSERT [INTO] tname}] }

   [ { RENAME {cname          } TO name                                         } ]
   [ {        {CONSTRAINT name}                                               } ]

   [ { REVALIDATE PRIMARY INDEX [WITH {DELETE | INSERT [INTO] tname}]          } ]
```

Any of the following options may be listed in any order:

```
   [NO] FALLBACK [PROTECTION]

   [NO  ]                    [ [NO         ]                ]
   [    ] [BEFORE] JOURNAL   [ [DUAL       ] AFTER JOURNAL  ]
   [DUAL]                    [ [[NOT] LOCAL]                ]

  WITH JOURNAL TABLE = [dbname.] tname

   [ FREESPACE = n [PERCENT] ]
   [ DEFAULT FREESPACE       ]

   [                    [ BYTES     ] ]
   [ DATABLOCKSIZE = n  [ KBYTES    ] ]
   [                    [ KILOBYTES ] ]
   [ { MINIMUM }                      ] [ IMMEDIATE ]
   [ { MAXIMUM } DATABLOCKSIZE        ]
   [ { DEFAULT }                      ]
```

The following options apply to Temporary tables only:

```
   [NO] LOG

   ON COMMIT { DELETE | PRESERVE } ROWS
```

```
ALTER TRIGGER [dbname.]name {DISABLED}
                            {ENABLED }
```

```
BEGIN INDEX ANALYSIS [ ON tname [..., tname] ] FROM workloadname INTO qcdname
   [ AS indextag ]
                { "Indexes Per Table"    }
                { "Tables Per Request"   }
                { "Search Space"         }
   [ SET MAXIMUM { "Change Rate"         } = n  [ ... ] ] ;
                { "Columns Per Index"    }
                { "NUSI Selectivity    " }
                { "VOSI Typical Percent" }
```

only INSERT EXPLAIN NEW INDEX *statements may be used here*

```
   END INDEX ANALYSIS;
```

```
BEGIN INSERT WORKLOAD INTO qcdname AS workloadname ;
```

only INSERT EXPLAIN *statements may be used here*

```
   END WORKLOAD;
```

```
                                          [FIRST         ] {ALL             }
{BEGIN} LOGGING [DENIALS] [WITH TEXT] ON  [LAST          ] {GRANT           }
{END  }                                   [FIRST AND LAST] {priv [... ,priv]}
                                          [EACH          ]

                                    { DATABASE dbname     }
                                    { USER     userid     }
   [ BY userid [... ,userid] ] [ ON { TABLE [dbname.]name } ]
                                    { VIEW  [dbname.]name }
                                    { MACRO [dbname.]name }
```

```
{BEGIN} QUERY LOGGING on ***TODO***
{ END }
```

```
{ BEGIN TRANSACTION }                                  { END TRANSACTION }
{ BT                } ; statement; [... statement;]    { ET              };
```

```
CALL [dbname.]tname ( [parm [(attributes)] [... ,parm [(attributes)] ] ] )
```

```
CHECKPOINT tname [ ,NAMED chkptname ]
```

```
COLLECT { STAT[ISTICS] | STATS } [ FOR SAMPLE percent [ PERCENT ] INTO qcdname ]

                              [ COLUMN cname                ]
        [ON] [TEMPORARY] tname [ INDEX  name                 ]
                              [ INDEX (cname [... ,cname] ) ]
```

```
             [ DATABASE ]
             [ USER     ]
             [ TABLE    ]           [ [ AS ]           ]
COMMENT [ON] [ VIEW     ] objname   [ [    ] 'string' ]
             [ MACRO    ]           [ [ IS ]           ]
             [ COLUMN   ]
             [ TRIGGER  ]
```

COMMIT [WORK]

```
{ CREATE DATABASE }
{                 } dbname [FROM ownerdb]
{ CD              }
```

```
AS PERM[ANENT] = n [BYTES]
```

```
[ [,] option [... [,] option] ]
```

Any of the following options may be listed in any order:

```
SPOOL = n [BYTES]
```

```
TEMPORARY = n [BYTES]
```

```
ACCOUNT = 'acctid'
```

```
[NO] FALLBACK [PROTECTION]
```

```
[NO  ]                    [ [NO         ]                ]
[    ] [BEFORE] JOURNAL   [ [DUAL       ] AFTER JOURNAL ]
[DUAL]                    [ [[NOT] LOCAL]                ]
```

```
DEFAULT JOURNAL TABLE = [dbname.]tname
```

CREATE [UNIQUE] **INDEX** [name] [ALL] (cname [... ,cname])

```
             {VALUES (cname)}
[ ORDER BY {HASH   (cname)} ]
             {       (cname)}
```

```
[ ... [UNIQUE] INDEX [name] (cname [... ,cname ] ) ]
```

```
ON [TEMPORARY] tname
```

CREATE HASH INDEX name (cname [... ,cname])

```
ON tname
```

```
BY (cname)
```

```
ORDER BY HASH (cname)
```

CREATE JOIN INDEX name [, FALLBACK] AS

```
          { { cname                                  }               }
          { { SUM     (cname) AS alias               }               }
SELECT    { { COUNT   (cname) AS alias               } [..., cname ] }
          { { EXTRACT (YEAR FROM datecol) AS alias }                 }
          { (cname [..., cname]) , (cname [..., cname])              }

          { tname [[AS] aname ] [..., tname [[AS] aname ]] }
          {                                                }
FROM      {       [        INNER  ]                        }
          { tname [ LEFT  [OUTER] ] JOIN tname [ON cond]   }
          {       [ RIGHT [OUTER] ]                        }
```

```
[ WHERE cond ]
```

```
[ GROUP BY (cname [..., cname]) ]
```

```
[ ORDER BY (cname) ]
```

```
[ PRIMARY INDEX [idxname] (cname [..., cname ] ) ] [ ORDER BY [HASH  ] [(cname)] ]
                                                             [VALUES]
```

```
[ INDEX [idxname] [ALL]   (cname [..., cname ] ) ] [ ORDER BY [HASH  ] [(cname)] ]
                                                             [VALUES]
```

```
{ CREATE MACRO }
              } macroname
{ CM          }

[ (pname datadesc [... ,pname datadesc ] ) ]

AS  ( [USING clause] [LOCKING clause] statement; [... statement; ] )
```

```
CREATE PROFILE name [ AS option [ ... ,option ] ]
```

Any of the following options may be listed in any order:

```
           { 'accountid'                          }
   ACCOUNT = { ('accountid' [ ... , 'account id' ]) }
           { NULL                                 }

   DEFAULT DATABASE = { databasename | NULL }

   SPOOL = { n [BYTES] | NULL }

   TEMPORARY = { n [BYTES] | NULL }

   PASSWORD [ATTRIBUTES] = { (attrib = val | NULL, [ ... ,attrib = val | NULL ]) | NULL }
```

*where **attrib** is one of the following, in any order:*

```
   EXPIRE = n              (0 to 32767)
   MINCHAR = n             (1 to 30)
   MAXCHAR = n             (1 to 30)
   DIGITS = Y | N
   SPECCHAR = Y | N
   MAXLOGONATTEMPTS = n    (0 to 256)
   LOCKEDUSEREXPIRE = n    (-1 to 32767)
   REUSE = n               (0 to 32767)
```

```
                           { [IN]  }                       { [IN]  }
CREATE PROCEDURE pname ( [ {  OUT  } vname vtype ] [... , {  OUT  } vname vtype] )
                           { INOUT }                       { INOUT }

   [label:] BEGIN [[NOT] ATOMIC]
      [ variable declarations ]
      [ error handler ]
      SPL Statement; [... , SPL Statement;]
   END [label] ;
```

```
CREATE ROLE rolename ;
```

```
      [SET     ] [VOLATILE        ]
CREATE [MULTISET] [GLOBAL TEMPORARY] TABLE tname [,option [... ,option] ]

   ( cname Datadesc [ColumnConstraint] [... , cname Datadesc [ColumnConstraint] ]
     [ , TableConstraint ] [... , TableConstraint]
   )

   [ [UNIQUE] PRIMARY INDEX [name] ( cname [... ,cname]) [PARTITION BY partexpr] ]

   [ ... [,] [UNIQUE] INDEX [name] ( cname [... ,cname] ) ]

                                                        {[VALUES]         }
   [ ... [,] INDEX [name] ( cname [... ,cname] ) ORDER BY {[HASH  ] (cname) } ]

   [ON COMMIT { DELETE   } ROWS ]
              { PRESERVE }                          ← Applies to Temporary tables only
```

Any of the following options may be listed in any order:

```
[NO] FALLBACK [PROTECTION]

[NO  ]                    [ [NO         ]                ]
[    ] [BEFORE] JOURNAL   [ [DUAL       ] AFTER JOURNAL ]
[DUAL]                    [ [[NOT] LOCAL]                ]

WITH JOURNAL TABLE = [dbname.]tname

FREESPACE = n [PERCENT]

[                    [ BYTES     ] ]
[ DATABLOCKSIZE = n  [ KBYTES    ] ]
[                    [ KILOBYTES ] ] [ IMMEDIATE ]
[ { MINIMUM }                      ]
[ { MAXIMUM } DATABLOCKSIZE        ]

[NO] LOG                                    ← Applies to Temporary tables only
```

TableConstraint *can be any of the following*:

```
[CONSTRAINT name] {PRIMARY KEY} (cname [... ,cname])
                  {UNIQUE     }

[CONSTRAINT name] CHECK (expr operator expr)

[CONSTRAINT name] FOREIGN KEY ( cname [... ,cname] )
                  REFERENCES [dbname.]tname [( cname [... ,cname] )]
```

ColumnConstraint *can be any of the following*:

```
[CONSTRAINT name] {PRIMARY KEY}
                  {UNIQUE     }

[CONSTRAINT name] CHECK (expr operator expr)

[CONSTRAINT name] REFERENCES [dbname.]tname [( cname [... ,cname] )]
```

Datadesc *consists of a Data Type and additional optional phrases*:

Data Types:

BYTEINT	SMALLINT	INTEGER	FLOAT [(n)]
DECIMAL(n [,m])	NUMERIC(n [,m])	DATE	REAL [(n)]
CHAR(n)	VARCHAR(n)	LONG VARCHAR	CHAR VARYING(n)
BYTE(n)	VARBYTE(n)	DOUBLE PRECISION	
GRAPHIC(n)	VARGRAPHIC(n)	LONG VARGRAPHIC	
TIMESTAMP[(n)]	TIME[(n)]	INTERVAL type[n] [TO type[n]]	

Optional phrases (general):

```
NOT NULL
FORMAT     'string'
TITLE      'string'
NAMED      name

COMPRESS  {value | NULL }
[ DEFAULT {value | NULL | USER | DATE | TIME} ]
[ WITH DEFAULT                                ]
```

Optional phrases (character columns only):

```
UPPERCASE | UC
[NOT] CASESPECIFIC | CS
CHARACTER SET { LATIN | UNICODE | KANJISJIS | GRAPHIC | KANJI1 }
```

```
       [SET     ] [VOLATILE        ]
CREATE [MULTISET] [GLOBAL TEMPORARY] TABLE tname [,option [... ,option] ]

   AS { [dbname.]tname     } WITH [NO] DATA
      { (select statement) }



                                {BEFORE    } {DELETE                          }
CREATE TRIGGER name [ENABLED ] {AFTER     } {INSERT                          }
                   [DISABLED] {INSTEAD OF} {UPDATE [OF (cname […, cname])] }

   ON [dbname.]tname [ORDER integer]

   [ [ REFERENCING OLD       [AS] name NEW       [AS] name ] FOR EACH ROW       ]
   [ [ REFERENCING OLD TABLE [AS] name NEW TABLE [AS] name ] FOR EACH STATEMENT ]

   [ WHEN condition ]

   ( command; [..., command;])



CREATE USER username [FROM ownerdb]

   AS PERM[ANENT] = n [BYTES] [,] PASSWORD = { NULL }
                                             { name }
   [ [,] option [... [,] option] ] ;
```

Any of the following options may be listed in any order:

```
   SPOOL = n [BYTES]

   TEMPORARY = n [BYTES]

   STARTUP = 'string;'

             { 'acctid'                    }
   ACCOUNT = {                             }
             { ('acctid' [... ,'acctid'] ) }

   DEFAULT DATABASE = dbname

   [NO] FALLBACK [PROTECTION]

   [NO  ]                    [ [NO         ]                ]
   [    ] [BEFORE] JOURNAL   [ [DUAL       ] AFTER JOURNAL ]
   [DUAL]                    [ [[NOT] LOCAL]                ]

   DEFAULT JOURNAL TABLE = [dbname.]tname

   COLLATION = { ASCII | EBCDIC | MULTINATIONAL | HOST | CHARSET_COLL | JIS_COLL }

   DATEFORM  = { ANSIDATE | INTEGERDATE }

   TIME ZONE = { LOCAL | NULL | [-] 'hh:mm' }

   DEFAULT CHARACTER SET { LATIN | UNICODE | KANJISJIS | GRAPHIC | KANJI1 }
{ CREATE VIEW }
              } viewname [ (cname [... ,cname] ) ] AS
{ CV          }

   [LOCKING clause] SELECT statement [GROUP BY clause] [HAVING clause]

   [WITH CHECK OPTION] [ORDER BY clause]
```

***Note** - no WITH clause is allowed on the select statement.*

```
DATABASE dbname
```

```
                                    [ WHERE cond ]
DEL[ETE] FROM tname [[AS] aname]]   [            ]
                                    [ ALL        ]
```

```
         { DATABASE }
DEL[ETE] {          } name [ALL]
         { USER     }
```

```
           {DUMP COSTS  sysname ['comment']       }
DIAGNOSTIC {HELP COSTS                            }
                       {  NOT  }        {REQUEST}
           {SET  COSTS {sysname} ON FOR {SESSION} }
                       {  TPA  }        {SYSTEM }
```

```
DIAGNOSTIC ValidateIndex ON FOR SESSION;
```

only CREATE INDEX *and* COLLECT STATISTICS *statements may be used here*

```
DIAGNOSTIC ValidateIndex NOT ON FOR SESSION;
```

```
     { DATABASE          }
     { JOIN INDEX        }
     { MACRO             }
DROP { PROCEDURE         } name
     { [TEMPORARY] TABLE }
     { TRIGGER           }
     { USER              }
     { VIEW              }
```

```
DROP INDEX [ALL] { ( cname [... ,cname ] ) }
                 { IdxName                 }

   [ ORDER BY {VALUES} ] ON [TEMPORARY] tname
              {HASH  }
```

```
                                                    [ COLUMN cname               ]
DROP STATISTICS [FROM qcdname] [ON] [TEMPORARY] tname [ INDEX  IdxName            ]
                                                    [ INDEX(cname [... ,cname] ) ]
```

```
     { 'string'  }
ECHO {           }
     { 'command' }
```

```
                    [ (expr [... ,expr ] )             ]
EXEC[UTE] macroname [                                  ]
                    [ (pname=expr [... ,pname=expr ] ) ]
```

```
GIVE name TO recipientname
```

```
      { ALL [PRIVILEGES]                   }
GRANT { privilege [... ,privilege ]        }
      { ALL BUT privilege [... ,privilege] }
```

```
          { dbname         }      { [ALL] name [... ,[ALL] name] }
      [ ON { dbname.objname } ] TO {                              }
          { objname        }      { PUBLIC                        }

      [WITH GRANT OPTION]
```

```
GRANT rolename [ ..., rolename ] TO user [ ..., user ] [WITH ADMIN OPTION]
```

```
      { MONITOR [ PRIVILEGES ]                   }
GRANT { monpriv [ ..., monpriv ]                 }
      { MONITOR BUT NOT monpriv [ ..., monpriv ] }

       TO [ALL] name [ ..., [ALL] name ] [WITH GRANT OPTION]
```

```
               { hostid [... ,hostid] }
GRANT LOGON ON {                      }
               {         ALL         }

      {        AS DEFAULT           }
      {                             } [WITH NULL PASSWORD]
      { TO username [... ,username] }
```

```
     {             COLUMN     cname [... ,cname] FROM tname [... ,tname ]         }
     {             COLUMN     * FROM [dbname.]tname [... ,tname]                  }
     {             COLUMN     [dbname.]tname.cname  [... ,tname.cname]            }
     {             COLUMN     [dbname.]tname.*                                    }
     {             CONSTRAINT [dbname.]tname                                      }
     {             DATABASE   dbname                                              }
     { [TEMPORARY] INDEX      [dbname.]tname [(cname [... ,cname]) ]              }
     { [TEMPORARY] INDEX      [dbname.]idxname                                    }
HELP {             JOIN INDEX [dbname.]idxname                                    }
     {             MACRO      [dbname.]macroname                                  }
     {             PROCEDURE  [dbname.]tname [ATTR[IBUTES]]                       }
     {             SESSION                                                        }
     {                                                 [FROM   qcdname          ] }
     { [TEMPORARY] STATISTICS [USING SAMPLE] tblname [COLUMN cname              ] }
     {                                                 [INDEX (cname [..., cname])] }
     {             TABLE      [dbname.]tblname                                    }
     {             TRIGGER    [dbname.]trigname                                   }
     {             VIEW       [dbname.]viewname                                   }
     {             VOLATILE   TABLE                                               }
     {             USER       username                                            }
```

```
       { ARCHIVE    }
       { BULKLOAD   }
       { DUMP       }
       { FASTLOAD   }
       { FASTEXPORT }
HELP ' { HELP       } [ CommandName ] '
       { MULTILOAD  }
       { PMPC       }
       { SPL        }
       { SQL        }
       { TPCCONS    }
```

```
                      {                        [VALUES] (expr [... ,expr] ) }
INS[ERT] [INTO] tname {   (cname [... ,cname]) VALUES  (expr [... ,expr] ) }
                      { [ (cname [... ,cname]) ] subquery                   }
                      { DEFAULT VALUES                                      }
```

MODIFY DATABASE dbname

```
AS option [... [,] option ] ;
```

Any of the following options may be listed in any order:

```
PERM[ANENT] = n [BYTES]

SPOOL = n [BYTES]

TEMPORARY = n [BYTES]

ACCOUNT = 'acctid'

[NO] FALLBACK [PROTECTION]

[NO  ]                   [ [NO          ]                ]
[    ] [BEFORE] JOURNAL  [ [DUAL        ] AFTER JOURNAL ]
[DUAL]                   [ [[NOT] LOCAL]                ]

[DEFAULT JOURNAL TABLE = [dbname.]tname ]
[                                       ]
[DROP DEFAULT JOURNAL TABLE [= tname]   ]
```

MODIFY PROFILE name [AS option [... ,option]]

Any of the following options may be listed in any order:

```
          { 'accountid'                           }
ACCOUNT = { ('accountid' [ ... , 'account id' ]) }
          { NULL                                  }

DEFAULT DATABASE = { databasename | NULL }

SPOOL = { n [BYTES] | NULL }

TEMPORARY = { n [BYTES] | NULL }

PASSWORD [ATTRIBUTES] = { (attrib = val | NULL, [ ... ,attrib = val | NULL ]) | NULL }
```

*where **attrib** is one of the following, in any order:*

```
EXPIRE = n              (0 to 32767)
MINCHAR = n             (1 to 30)
MAXCHAR = n             (1 to 30)
DIGITS = Y | N
SPECCHAR = Y | N
MAXLOGONATTEMPTS = n    (0 to 256)
LOCKEDUSEREXPIRE = n    (-1 to 32767)
REUSE = n               (0 to 32767)
```

```
MODIFY USER username

   AS option [... [,] option ] ;
```

Any of the following options may be listed in any order:

```
   PERM[ANENT] = n [BYTES]

   PASSWORD = { name | NULL }

   SPOOL = n [BYTES]

   TEMPORARY = n [BYTES]

   STARTUP = { 'string' | NULL }

             { 'acctid'                    }
   ACCOUNT = {                             }
             { ('acctid' [... ,'acctid'] ) }

   DEFAULT DATABASE = dbname

   [NO] FALLBACK [PROTECTION]

   [NO  ]                    [ [NO         ]                ]
   [    ] [BEFORE] JOURNAL   [ [DUAL       ] AFTER JOURNAL ]
   [DUAL]                    [ [[NOT] LOCAL]                ]

   [DEFAULT JOURNAL TABLE = [dbname.]tname ]
   [                                       ]
   [DROP DEFAULT JOURNAL TABLE [= tname]   ]

   RELEASE PASSWORD LOCK

   COLLATION = { ASCII | EBCDIC | MULTINATIONAL | HOST | CHARSET_COLL | JIS_COLL }

   DATEFORM  = { ANSIDATE | INTEGERDATE }

   TIME ZONE = { LOCAL | NULL | [-] 'hh:mm' }

   DEFAULT CHARACTER SET { LATIN | UNICODE | KANJISJIS | GRAPHIC | KANJI1 }



       { MACRO      }
RENAME { PROCEDURE  } oldname { TO } newname
       { TABLE      }         { AS }
       { TRIGGER    }
       { VIEW       }


REPLACE MACRO macroname

   [ (pname datadesc [... ,pname datadesc ] ) ]

   AS  ( [USING clause] [LOCKING clause] statement; [... statement; ] )
```

```
                              { [IN]  }                       { [IN]  }
REPLACE PROCEDURE pname ( [ [ {  OUT  } vname vtype ] [... , [ {  OUT  } vname vtype] ] )
                              { INOUT }                       { INOUT }

   [label:] BEGIN [[NOT] ATOMIC]
      SPL Statement; [... , SPL Statement;]
   END [label:] ;


                                  { BEFORE     } { DELETE                          }
REPLACE TRIGGER name [ENABLED ] [ { AFTER      } { INSERT                          }
                    [DISABLED] [ { INSTEAD OF } { UPDATE [OF cname [..., cname]] } ]

   ON [dbname.]tname  [ ORDER integer ]

   [ [ REFERENCING OLD       [AS] name NEW       [AS] name ] FOR EACH ROW       ]
   [ [ REFERENCING OLD TABLE [AS] name NEW TABLE [AS] name ] FOR EACH STATEMENT ]

   [ WHEN condition ]

   ( command; [..., command;])


REPLACE VIEW  viewname  [ (cname [... ,cname] ) ] AS

   [LOCKING clause] SELECT statement

   [WITH CHECK OPTION]

   Note - no WITH clause is allowed on the select statement.


                          { ALL [PRIVILEGES]                    }
REVOKE [GRANT OPTION FOR] { privilege [... ,privilege]          }
                          { ALL BUT privilege [... ,privilege] }

      { dbname         } {FROM} { [ALL] name [... ,[ALL] name] }
   ON { dbname.objname } {    } {                              }
      { objname        } { TO } { PUBLIC                        }


                          { MONITOR [ PRIVILEGES ]                    }
REVOKE [GRANT OPTION FOR] { monpriv [ ..., monpriv ]                  }
                          { MONITOR BUT NOT monpriv [ ..., monpriv ] }

      { FROM } [ALL] name [ ..., [ALL] name ]
      {  TO  }


               { hostid [... ,hostid] } { AS DEFAULT              }
REVOKE LOGON ON {                      } { {FROM} name [... ,name] }
               {         ALL         } { { TO }                   }


ROLLBACK [WORK] ['msgtext'] [FROM tname [ ..., tname ]] [WHERE cond ] ;
```

```
         [ALL     ] { *                                          }
SEL[ECT] [        ] { expr [[AS] name] [... ,expr [[AS] name] ] }
         [DISTINCT] { tname.*          [... ,tname.*          ] }

        [ tname [[AS] aname] ]                          ]
        [                    ]                          ]
        [       {       [INNER] }                       ]
        [       { LEFT  [OUTER] }                       ]
   FROM [ tname { RIGHT [OUTER] } JOIN tname ON [cond] ] [... ,tname ... ]
        [       { FULL  [OUTER] }                       ]
        [                    ]                          ]
        [ tname CROSS JOIN tname                        ]
        [                    ]                          ]
        [ (subquery) [AS] aname ([cname] [... ,cname]) ]

          { { expr            } {operator} [ANY ]          }
          { {                 } {IN      } [SOME] subquery }
          { {(expr [...,expr])} {NOT IN  } [ALL ]          } [... OR  cond]
   [WHERE {                                                }                ]
          { [NOT] EXISTS subquery                           } [... AND cond]
          {                                                 }
          { comparison                                      }

            { cname   }        { cname   }
   [GROUP BY {        } [... , {        } ] ]
            { col-pos }        { col-pos }

            { cname   } [ASC ]        { cname  } [ASC ]
   [ORDER BY {        } [    ]  [... ,{       } [    ] ] ]
            { col-pos } [DESC]        { col-pos} [DESC]

   [HAVING   cond ]

   [QUALIFY  cond ]

   [SAMPLE   n [... , n] ]

   [WITH expr [... , expr] [ BY expr [... , expr] ] ]
```

SET SESSION ACCOUNT 'acct-id' FOR { SESSION | REQUEST }

```
                          [ ASCII         ]
{ SET SESSION }           [ EBCDIC        ]
{             } COLLATION [ MULTINATIONAL ]
{ SS          }           [ HOST          ]
                          [ CHARSET_COLL  ]
                          [ JIS_COLL      ]
```

SET SESSION DATEFORM = { ANSIDATE | INTEGERDATE }

```
                       { LATIN1    }
SET SESSION EXPORT FOR { UNICODE   } { DEFAULT }
                       { KANJISJIS } { n       }
                       { GRAPHIC   }
```

SET SESSION OVERRIDE REPLICATION { ON | OFF }

```
             { LOCAL                                }
SET TIME ZONE { USER                                }
             { INTERVAL [-] 'hh:mm' HOUR TO MINUTE }
```

```
     { JOIN INDEX }
     { MACRO      }
SHOW { PROCEDURE  } name
     { TABLE      }
     { TRIGGER    }
     { VIEW       }
```

```
SHOW  dml-statement
```

```
UPD[ATE] tname [[AS] aname]

   [ FROM  tname [[AS] aname] [... ,tname [[AS] aname] ] ]

  SET  cname = expr [... , cname = expr]

   [ WHERE cond ]
   [            ] ;
   [ ALL        ]
```

```
UPDATE tname

    SET cname = expr [... , cname = expr]

  [ WHERE cond ]

ELSE

INSERT INTO tname [ (colname [... , colname] ) ]

    VALUES (expr [... , expr])
```

SQL STATEMENT MODIFIERS

The following modifiers can be used with any SQL statement.

Note - To use ROW locking, the statement must be a SELECT statement that uses a UNIQUE index. (Preparing for an UPDATE of that row)

```
EXPLAIN statement
```

```
DUMP EXPLAIN    INTO qcdname [ AS queryname ] statement
```

```
INSERT EXPLAIN [WITH [NO] STAT[ISTICS] [AND DEMOGRAPHICS] FOR (tname [..., tname]]) ]
               INTO (qcdname) [ AS (queryname) ]
               [ LIMIT [ SQL [ = n] ] ]
               [ FOR (frequency) ] statement
```

```
INSERT EXPLAIN NEW INDEX { FOR queryid | ALL }
```
<== only valid after BEGIN INDEX ANALYSIS

```
                                      { ACCESS      }
          { [DATABASE] dbname } [FOR] { EXCL[USIVE] }
LOCK[ING] { [TABLE] tname     } [   ] { SHARE       } [MODE] [NOWAIT] statement
          { [VIEW] vname      } [IN ] { READ        }
          { ROW               }       { WRITE       }
```

```
USING ( name datadesc  [... ,name datadesc ] ) request
```

STORED PROCEDURE LANGUAGE

SPL statements are listed alphabetically. Variables used within SQL statements must be prefixed by a semi-colon.

The following system variables may be referenced:

SQLCODE **SQLSTATE** **ACTIVITY_COUNT**

```
CALL [dbname.]procname (arg [... ,arg])
```

```
CASE  expr
    WHEN val THEN
         statement; [... statement;]
  [ WHEN val THEN
         statement; [... statement;] ] ...
  [ ELSE
         statement; [... statement;] ]
END CASE ;
```

```
CASE
    WHEN cond THEN
         statement; [... statement;]
  [ WHEN cond THEN
         statement; [... statement;] ] ...
  [ ELSE
         statement; [... statement;] ]
END CASE ;
```

```
CLOSE CURSOR cursorname ;
```

```
DECLARE {CONTINUE} HANDLER FOR {SQLSTATE 'nnnnn' [... ,SQLSTATE 'nnnnn']}
        {  EXIT  }             {SQLEXCEPTION                            }
                               {NOT FOUND                               }

   { statement                          }
   { BEGIN statement; [... statement;] END} ;
```

```
DECLARE varname [... ,varname] datatype [DEFAULT literal | NULL] ;
```

```
DECLARE cursorname [[NO]SCROLL] CURSOR FOR
   SelectStatement
   [ FOR {READ ONLY | UPDATE} ] ;
```

```
FETCH [FIRST | NEXT] FROM cursorname INTO var1 [... ,varn] ;
```

```
[label:] FOR var AS [ cursorname CURSOR FOR ] selectstatment DO
   statement; [... statement;]
END FOR [label] ;
```

```
IF cond THEN
   statement; [... statement;]
[ ELSEIF cond THEN
   statement; [... statement;] ] ...
[ ELSE
   statement; [... statement;] ]
END IF ;
```

```
ITERATE label ;
```

```
LEAVE label ;
```

```
[label:] LOOP
   statement; [... statement;]
END LOOP [label] ;
```

```
OPEN cursorname ;
```

```
PRINT { varname } [... , { varname } ] ;
      { literal }        { literal }
```

```
REPEAT
   statement; [ ... statement;]
   UNTIL cond
END REPEAT;
```

```
SELECT   expr [... , expr]
   INTO  :var [... , :var]
   FROM  tname
   WHERE cond
```

```
SET varname = expression ;
```

```
[label:] WHILE cond DO
   statement; [... ,statement;]
END WHILE [label] ;
```

BTEQ COMMANDS

BTEQ commands are listed alphabetically. Defaults are underscored.
Quoted strings may use either single (') or double (") quote marks.

```
.ABORT
```

```
         [   SUBSET    ]
.CMS     [             ]
         [ cms-command ]
```

```
         { FILE }     {'filename'}        {  SPL  }
.COMPILE {  DD  } [=] { filename } [ WITH { PRINT } ]
         {DDNAME}     {"filename"}        { NOSPL }
                                          {NOPRINT}
```

```
         [     n       ]
.EXIT    [ ERRORLEVEL  ]
         [ ERRORCODE   ]
         [ ACTIVITYCO  ]
```

```
         { DATA             }
.EXPORT  { INDICDATA        } {DDNAME} = name [,LIMIT=n] [,{CLOSE} ]
         { REPORT           } { FILE }                   [ {OPEN } ]
         { REPORTWIDE       }
         { DIF [DATALABELS] }                [AXSMOD [name] ['init str']]
```

```
.EXPORT  RESET
```

```
= [ n ]
```

```
.GOTO    labelname
```

```
.HANG    [ n ]
```

```
.HELP    BTEQ
```

```
         {   ERRORCODE   }                 { BTEQ command }
.IF      {   ERRORLEVEL  } operator n THEN {              }
         { ACTIVITYCOUNT }                 { SQL request  }
```

```
         {   DATA    } {  FILE  }
.IMPORT  {           } { DDNAME } = name [,SKIP=n]          ← Channel client
         { INDICDATA }
```

```
        { DATA          }
.IMPORT { INDICDATA     } {DDNAME} { name } [,SKIP=n] [AXSMOD [name] ['init str']]
        { REPORT        } { FILE } {'name'}
        { VARTEXT ['c'] }          {"name"}
```

```
.LABEL  labelname
```

```
.LOGOFF
```

```
.LOGON  [tdpid /] username [,password [,'acctid'] ]
```

```
            {  file  }
.MESSAGEOUT {        } = name
            { ddname }
```

```
.OS command
```

```
        [     n      ]
.QUIT   [ ERRORLEVEL ]
        [ ERRORCODE  ]
        [ ACTIVITYCO ]
```

```
.REMARK 'string [ //string [... //string ] ]'   ← Can use single or double quotes
```

```
        [ n ]
.REPEAT [   ]
        [ * ]
```

```
        {   DD   }   { 'name' }
.RUN    { DDNAME } = {  name  } [,SKIP=n]
        {  FILE  }   { "name" }
```

```
.[SET] DEFAULTS
```

```
                  [ OFF ]
.[SET] ECHOREQ    [     ]
                  [ ON  ]
```

```
                  { { UNKNOWN        }                        }
.[SET] ERRORLEVEL { { nnnn           } SEVERITY nn [ ... ,] }
                  { {(nnnn ... ,nnnn)}                        }
                  {  ON                                       }
                  {  OFF                                      }
```

```
                [ STDOUT ]
.[SET] ERROROUT [        ]
                [ STDERR ]


                [ OFF ] [ n [... ,n] ]
.[SET] FOLDLINE [     ] [            ]
                [ ON  ] [ ALL        ]


.[SET] FOOTING   'string [ //string [... //string ] ]'

                [ OFF ]
.[SET] FORMAT   [     ]
                [ ON  ]


                [      ON       ]
.[SET] FORMCHAR [      OFF      ]
                [ 'hexstring'xb ]


                [ OFF ]
.[SET] FULLYEAR [     ]
                [ ON  ]


.[SET] HEADING   'string [ //string [... //string ] ]'


                [ OFF]
.[SET] INDICDATA [    ]
                [ ON ]


                  [ ON  ]
.[SET] LOGONPROMPT [     ]
                  [ OFF ]


.[SET] MAXERROR  nn


                [  OFF   ] [ EXIT   name     ]
.[SET] NOTIFY   [  LOW   ] [ MSG   [text]    ]
                [ MEDIUM ] [ QUEUE [options] ]
                [  HIGH  ]


.[SET] NULL [AS] 'string'


                [ OFF ] [ n [..., n] ]
.[SET] OMIT     [     ] [            ]
                [ ON  ] [ ALL        ]


                [ OFF ] [ n [,n ,n ...] ]
.[SET] PAGEBREAK [     ] [              ]
                [ ON  ] [ ALL           ]
```

```
.[SET] PAGELENGTH n
```

```
                  [ OFF ]
.[SET] QUIET      [     ]
                  [ ON  ]
```

```
                  [ OFF ]
.[SET] RECORDMORE [     ]
                  [ ON  ]
```

```
                  [ OFF ]
.[SET] REPEATSTOP [     ]
                  [ ON  ]
```

```
                  [ OFF ]
.[SET] RETCANCEL  [     ]
                  [ ON  ]
```

```
.[SET] RETLIMIT   n
```

```
                  [ OFF ]
.[SET] RETRY      [     ]
                  [ ON  ]
```

```
.[SET] RTITLE     'string [ //string [... //string ] ]'
```

```
                  {     ALL     }
.[SET] SECURITY   { PASSWORD[S] }
                  {     NONE    }
```

```
                  [ 'string' ]
.[SET] SEPARATOR  [ "string" ] [ALL]
                  [     n    ]
```

```
                        {  charsetnum   }
.[SET] SESSION CHARSET  { 'charsetname' }
                        { "charsetname" }
```

```
                        [ NONE         ]
.[SET] SESSION SQLFLAG  [ INTERMEDIATE ]
                        [ ENTRY        ]
```

```
                                { BTET }
.[SET] SESSION TRANS[ACTION]    {      }
                                { ANSI }
```

```
.[SET] SESSIONS    n


                   [ OFF ] [ 0            ]
.[SET] SIDETITLES  [     ] [ wn [... ,wn] ]
                   [ ON  ] [ ALL          ]


                   [ OFF ] [ n [... ,n] ]
.[SET] SKIPDOUBLE  [     ] [             ]
                   [ ON  ] [ ALL         ]


                   [ OFF ] [ n [... ,n] ]
.[SET] SKIPLINE    [     ] [             ]
                   [ ON  ] [ ALL         ]


                   [ OFF ] [ n [... ,n] ]
.[SET] SUPPRESS    [     ] [             ]
                   [ ON  ] [ ALL         ]


.[SET] TDP         TDPn


                   [ DEFAULT ]
.[SET] TIMEMSG     [         ]
                   [ QUERY   ]


                   [ OFF ] [ 0            ]
.[SET] TITLEDASHES [     ] [ wn [... ,wn] ]
                   [ ON  ] [ ALL          ]


                   [ OFF ] [ n [... ,n] ]
.[SET] UNDERLINE   [     ] [             ]
                   [ ON  ] [ ALL         ]


.[SET] WIDTH       n


.SHOW CONTROL[S]


.SHOW ERRORMAP


.SHOW VERSION[S]


.TDP xx[xxxxxx]                (Optional form for VM users only)


.TSO string
```

PREPROCESSOR2 STATEMENTS

Preprocessor2 statements are shown below in alphabetical order.
In addition to the DML statement variations shown here you may also use other DML and DDL statements described in the DBC Reference Manual.

Each statement must be prefixed by 'EXEC SQL' and followed by the statement terminator. ('END-EXEC' in Cobol, or ';' in PL/1 or C.)

```
BEGIN DECLARE SECTION
      .
   < Variable Definitions >
      .
   END DECLARE SECTION
```

Note: Use in 'C' programs only.

```
                            [       {ckpt-label}]
CHECKPOINT [dbname.]tbname [,NAMED {           }] INTO [:]host-variable
                            [       {:labelvar }]

   [[INDICATOR] :host-variable]
```

```
CLOSE cursor-name
```

```
COMMENT [ON] objkind objref [IS] 'comment'
                            [AS]
```

```
COMMENT [ON] objkind objref INTO [:]host-variable [[INDICATOR] :host-variable]
```

```
COMMIT [WORK [RELEASE] ]
```

```
CONNECT [:]id-var IDENTIFIED BY [:]password-var
```

```
         {dbname    }
DATABASE {          }
         {:dbnamevar}
```

```
                                {cursor-specification            }
DECLARE cursor-name CURSOR FOR {statement-name                   }
                               {'request-specification'          }
                               {EXEC [dbname.]macroname[(parms)]}
```

```
DECLARE statement-name [ ... , statement-name] STATEMENT
```

```
        { Table-name }
DECLARE {            } TABLE ( column-spec [ ... , column-spec] )
        { View-name  }
```

```
DEL[ETE] FROM tbl-name WHERE CURRENT OF cursor-name
```

```
                                                [       {NAMES } ]
DESCRIBE statement-name INTO [:]descriptor-area [ USING {ANY   } ]
                                                [       {BOTH  } ]
                                                [       {LABELS} ]

         [               {stmt-number} ]
         [ FOR STATEMENT {           } ]
         [               {[:]numvar  } ]
```

```
EXEC [dbname.] macroname [ (parm-list) ]
```

```
                       [       {[:]h-var [[INDICATOR] :h-var] [... , h-var…] } ]
EXECUTE statement-name [ USING {                                              } ]
                       [       {DESCRIPTOR [:]descriptor-area                 } ]
```

```
                  {statement-string  }
EXECUTE IMMEDIATE {                  }
                  {[:]stmt-string-var}
```

```
                  { INTO [:]host-var [[INDICATOR] :host-var] [... , host-var…] }
FETCH cursor-name {                                                            }
                  { USING DESCRIPTOR [:]descriptor-area                        }
```

```
        {SQLCA    }
INCLUDE {SQLDA    }
        {text-name}
```

```
INS[ERT] [INTO] tname [ (cname [... ,cname]) ] VALUES ( [:]h-var [... , [:]h-var] )
```

```
LOGON [:]logonstr
```

```
                [       { [:]h-var [[INDICATOR] :h-var] [... , h-var…] } ]
OPEN cursor-name [ USING {                                             } ]
                [       { DESCRIPTOR [:]descriptor-area                } ]
```

```
                    [ TO NEXT                       ]
POSITION cursor-name [ TO [STATEMENT] {stmt-number} ]
                    [                 {[:]numvar  } ]
```

```
PREPARE statement-name

                             [       {NAMES }]
    [ INTO [:]descriptor-area [ USING {ANY   }] [ FOR STATEMENT {stmt-number} ] ]
```

```
                              [       {BOTH  }]                 {[:]numvar  }
                              [       {LABELS}]

    FROM {statement-string  }
         {[:]stmt-string-var}
```

REWIND cursor-name

ROLLBACK [WORK [RELEASE]] [abort-message] [WHERE abort-cond]

```
SEL[ECT] [ALL     ] expr [... , expr]
         [DISTINCT]

    INTO [:]host-var [[INDICATOR] :host-var] [... , host-var]

    from-clause

    [where-clause]
```

SET BUFFERSIZE size

```
            { set-name      }
SET CHARSET {               }
            { :set-name-var }
```

```
     {MACRO}
SHOW {TABLE} [dbname.]obj-name INTO [:]host-variable [[INDICATOR] :host-variable]
     {VIEW }
```

UPD[ATE] [dbname.]tbl-name [alias-name] SET col-name = expr [... ,col-name = expr]

    WHERE CURRENT OF cursor-name

```
         { SQLERROR   } { CONTINUE       }
WHENEVER { SQLWARNING } { GO TO [:]label }
         { NOT FOUND  } { GOTO  [:]label }
                        { PERFORM code   }      ← COBOL Only
                        { CALL  function }
```

ARCHIVE / RECOVERY COMMANDS

This section summarizes the syntax used by the Archive and Recovery utility. Statements are listed alphabetically.

(Note - The Keyword **DDNAME** may be used in place of **FILE** on MVS & VM)

```
        { *                                                    }
        { ALL                                                  }
ANALYZE { { ( databasename )                          }        }  [ CATALOG ]
        { { ( databasename1 ) TO ( databasename2 )} [ ,... ] }

       { DISPLAY [LONG] }
    [, {                } ]
       { VALIDATE       }

    [, USE {ASCII } COLLATION ]
           {EBCDIC}

    , FILE = name
```

```
        {   DATA    }
ARCHIVE {DICTIONARY } TABLE[S]
        {NO FALLBACK}
        {  JOURNAL  }

   { (databasename)     [ (EXCLUDE TABLES (tblname  [..., tblname]  )) ] }
   { (databasename) ALL [ (EXCLUDE TABLES (db.tname [..., db.tname] )) ] } [, ...]
   { (databasename.tablename)                                             }

   [, option [ ... ,option ] ]

    ,  FILE = name [, FILE = name]
```

Any of the following options may be listed:

```
             { ( databasename ) [ ALL ]               }
    EXCLUDE  {                                        } [, ...]
             { ( databasename1 ) TO ( databasename2 ) }

    { PN  = ccc-p [... , ccc-p ]   }    ← V1 systems only
    { AMP = n     [... , n ]       }    ← V2 systems only
    { CLUSTER[S] = nnn [... , nnn] }

    RELEASE LOCK

    INDEXES

    ABORT

               { READ }
    USE [GROUP] {      }
               { LOCK }

    NONEMPTY DATABASE[S]
```

```
       [   DATA TABLES      ] { ( databasename ) [ALL]     }
BUILD  [   JOURNAL TABLES   ] {                            } [,...]
       [NO FALLBACK TABLE[S]] { ( databasename.tablename ) }

               { ( databasename ) [ ALL ]              }
    [, EXCLUDE {                                       } [,...] ]
               { ( databasename1) TO ( databasename2 ) }

    [, RELEASE LOCK ]

    [, ABORT ]



            { ( databasename ) [ALL]     }
CHECKPOINT  {                            } [ ,...]
            { ( databasename.tablename ) }

               { ( databasename ) [ ALL ]              }
    [, EXCLUDE  {                                      } [ ,...] ]
               { ( databasename1 ) TO ( databasename2 )}

    [, WITH SAVE ]

    [      {ACCESS}      ]
    [, USE {      } LOCK ]
    [      { READ }      ]

    [, NAMED chkptname]



     {    DATA     }          { ( databasename )           }
COPY { DICTIONARY  } TABLE[S] {                            }
     {   JOURNAL   }          { ( databasename.tablename ) }
     { NO FALLBACK }

    [, option [ ... , option ] ] , FILE = name;
```

Any of the following options may be entered:

```
    ( [ FROM { ( databasename           ) } ]
      [      { ( databasename.tablename ) } ]
      [ , NO FALLBACK  ]
      [   { NO JOURNAL                                      } ]
      [ , { WITH JOURNAL TABLE = db.tablename               } ]
      [   { APPLY TO (db.tablename) [..., (db.tablename) ]  } ]
    )

    { PN  = ccc-p [... , ccc-p ]   }    ← V1 systems only
    { AMP = n     [... , n ]       }    ← V2 systems only
    { CLUSTER[S] = nnn [... , nnn] }

    NO BUILD

    RELEASE LOCK

    ABORT

    USE {ASCII } COLLATION
        {EBCDIC}

    REPLACE CREATORNAME
```

```
DELETE DATABASE ( databasename ) [ALL] [, ...]

               { ( databasename  ) [ALL]                }
    [, EXCLUDE {                                        } [,...] ]
               { ( databasename1 ) TO ( databasename2 ) }


        {SAVED   }         { ( databasename ) [ALL]     }
DELETE  {        } JOURNAL {                            } [,...]
        {RESTORED}         { ( databasename.tablename ) }

               { ( databasename ) [ALL]                 }
    [, EXCLUDE {                                        } [,...] ]
               { ( databasename1 ) TO ( databasename2 ) }


LOGOFF


LOGON [tdpid/] username , password [ , 'accid' ]


             { (databasename) [ALL]     }
RELEASE LOCK {                          } [, ...]
             { (databasename.tablename) }

    [, option [... , option ] ] ;
```

Any of the following options may be entered:

```
             { (databasename) [ALL]               }
    EXCLUDE  {                                    } [, ...]
             { (databasename1) TO (databasename2) }

    { PN  = ccc-p [... , ccc-p ]   }    ← V1 systems only
    { AMP = n     [... , n ]       }    ← V2 systems only
    { CLUSTER[S] = nnn [... , nnn] }

    ALL

    OVERRIDE

    BACKUP NOT DOWN
```

```
        {   DATA    }          { (databasename) [ALL]     }
RESTORE {DICTIONARY } TABLE[S] {                          } [,...]
        {NO FALLBACK}          { (databasename.tablename) }
        {  JOURNAL  }

    [, option [... , option] ]

    , FILE = filename
```

Any of the following options may be entered:

```
            { ( databasename ) [ALL]                 }
    EXCLUDE {                                        } [,...]
            { ( databasename1 ) TO ( databasename2 ) }

    { PN  = ccc-p [... , ccc-p ]   }    ← V1 systems only
    { AMP = n     [... , n ]       }    ← V2 systems only
    { CLUSTER[S] = nnn [... , nnn] }

    RESTORE FALLBACK

    NO BUILD

    RELEASE LOCK

    ABORT

    USE {ASCII } COLLATION
        {EBCDIC}
```

```
                           { ( databasename ) [ALL]     }
 REVALIDATE REFERENCES FOR {                            } [,...]
                           { ( databasename.tablename ) }

              { ( databasename  ) [ALL]                 }
    [, EXCLUDE {                                        } [,...] ]
              { ( databasename1 ) TO ( databasename2 ) }

    [, RELEASE LOCK ]

    [, ERRORDB = dbname ]
```

```
          { ( databasename ) [ALL]     }
ROLLBACK  {                            } [,...]
          { ( databasename.tablename ) }

    [, option [... , [ option ] ]

          { CURRENT  }
    , USE {          } JOURNAL
          { RESTORED }
```

Any of the following options may be entered:

```
            { ( databasename ) [ALL]                 }
    EXCLUDE {                                        } [,...]
            { ( databasename1 ) TO ( databasename2 ) }

       { chkptname          }
    TO { chkptname, eventno }
       { eventno            }

    { PN = ccc-p [... , ccc-p] }     ← V1 systems only
    { AMP = n    [... , n]     }     ← V2 systems only

    RELEASE LOCK

    [NO] DELETE

    ABORT
```

```
             { ( databasename ) [ALL]     }
ROLLFORWARD  {                            } [,...]
             { ( databasename.tablename ) }

    [, option [... , [ option ] ]

          { CURRENT  }
    , USE {          } JOURNAL
          { RESTORED }
```

Any of the following options may be entered:

```
             { ( databasename ) [ALL]                 }
    EXCLUDE  {                                        } [,...]
             { ( databasename1 ) TO ( databasename2 ) }

       { chkptname          }
    TO { chkptname, eventno }
       { eventno            }

    { PN = ccc-p [... , ccc-p] }     ← V1 systems only
    { AMP = n    [... , n]     }     ← V2 systems only

    PRIMARY DATA

    RELEASE LOCK

    [NO] DELETE

    ABORT
```

FASTLOAD COMMANDS

This section summarizes the command syntax used by the Fastload utility. All statements are listed alphabetically.

```
BEGIN    LOADING    [dbname.]tblname

         ERRORFILES [dbname.]tblname , [dbname.]tblname

         [CHECKPOINT n]

         [INDICATORS  ]
```

```
CLEAR
```

```
DEF[INE] [     fldname ( datatype [ ,NULLIF [=] value ] )
         [... ,fldname ( datatype [ ,NULLIF [=] value ] ) ] ]

         [ { DDNAME = filename } ]
         [ { FILE   = filename } ]
         [ { INMOD  = mod-name } ]
```

```
END LOADING
```

```
ERRLIMIT rows
```

```
HELP     [ TABLE [dbname.]tblname ]
```

```
INS[ERT] [INTO] [dbname.]tblname.*
```

```
INS[ERT] [INTO] [dbname.]tblname [ (cname [... ,cname] ) ]

         VALUES ( :fldname [... , :fldname] )
```

```
LOGOFF
```

```
LOGON    [tdpid /] username , password [ , 'acctid' ]
```

```
OS       command
```

```
QUIT
```

```
RECORD   [startnum] [THRU endnum]


SESSIONS n


SET RECORD [ { FORMATTED } ]              ← Network attached systems only
           [ {UNFORMATTED} ]


                     {    ASCII     }
SET SESSION CHARSET  { KANJIEUC_0U  }     ← Network attached systems only
                     { KANJISJIS_0S }
                     {      n       }


SHOW [VERSION[S]]
```

The following SQL statements are also supported by Fastload:

```
CREATE TABLE


DATABASE


DELETE


DROP TABLE
```

The following parameters are supported by the fastload command:

On Channel attached systems	On network attached systems
BUFSIZE = n	-b n
CHARSET = char-set-name	-c char-set-name
ERRLOG = filename	-e filename
TENACITY = hours	-t hours
-s mins	
INMODTYPE = SAS_C	

MULTILOAD COMMANDS

This section summarizes the command syntax used by the Multiload utility. All statements are listed alphabetically.

```
                                                          {pos1          }
.ACCEPT var [... ,var] [FROM] FILE fileid [ IGNORE {pos1 THRU     } ]
                                                          {THRU pos2     }
                                                          {pos1 THRU pos2}
```

```
.BEGIN  DELETE  MLOAD TABLES [dbname.]tname [... ,[dbname.]tname]

              [ WORKTABLES   [dbname.]tname [... ,[dbname.]tname] ]

              [ ERRORTABLES  [dbname.]tname [... , [dbname.]tname] ]

              [ TENACITY  hours ]

              [ SLEEP     mins  ]

                        { OFF    } { MSG   text   }
              [ NOTIFY  { LOW    } { EXIT  name   } ]
                        { MEDIUM } { QUEUE option }
                        { HIGH   }
```

```
.BEGIN [IMPORT] MLOAD TABLES [dbname.]tname [... , [dbname.]tname]

              [ WORKTABLES   [dbname.]tname [... , [dbname.]tname] ]

              [ ERRORTABLES  [dbname.]tname [... , [dbname.]tname] ]

              [ ERRORLIMIT  errcount [errpercent] ]

              [ CHECKPOINT  rate ]

              [ SESSIONS    limit ]

              [ TENACITY    hours ]

              [ SLEEP       mins  ]

                          { NONE  }
              [ AMPCHECK  { APPLY } ]
                          { ALL   }

                          { OFF    } { MSG   text   }
              [ NOTIFY    { LOW    } { EXIT  name   } ]
                          { MEDIUM } { QUEUE option }
                          { HIGH   }
```

```
 DEL[ETE] [FROM] [dbname.]tblname

          WHERE  colname = :fldname [{AND} colname = :fldname]
                                    {OR }
```

```
.DISPLAY 'text' [TO] FILE fileid
```

```
                  { { MARK   } DUPLICATE [{INSERT}] }
                  { { IGNORE }           [{UPDATE}] }
                  {                      [{DELETE}] }
.DML LABEL label [ { { MARK   } MISSING   [{UPDATE}] } ROWS ]
                  { { IGNORE }           [{DELETE}] }
                  {                      [{INSERT}] }
                  { DO INSERT FOR [MISSING UPDATE]  }
```

```
.END MLOAD ;
```

```
.FIELD fldname { startpos datadesc } [ NULLIF nullexpr ]
               { fieldexpr         }

        [ DROP {LEADING } {NULLS } [ [AND] {TRAILING} {NULLS } ] ]
               {TRAILING} {BLANKS}         {LEADING } {BLANKS}
```

```
.FILLER [fldname] startpos datadesc
```

```
.IF conditional-expression [THEN] ;
    statement 1
        ...
    statement n
[
.ELSE ;
    statement 1
        ...
    statement n
]
.ENDIF
```

```
.IMPORT   {  INFILE filename { FREE }                       }
          {                  { HOLD }                       }
          { [INFILE filename] INMOD modname [USING (parms)] }

          [ FROM m ] [ { FOR  n } ] LAYOUT layoutname
                       { THRU n }

          [ APPLY label [ WHERE condition ] ] [ APPLY ... ]
```

```
 INSERT   INTO [dbname.]tblname {.*                              }
                                {VALUES (:fldname [... ,fldname])}
```

```
.LAYOUT   layoutname [ CONTINUEIF condition ] [ INDICATORS ]
```

```
.LOGOFF   [retcode]
```

```
.LOGON    [tdpid /] username [,password [,'acctid']]
```

```
.LOGTABLE [dbname.]tablename
```

```
                { [TO] FILE fileid                                    }
.ROUTE MESSAGES {                  [WITH] ECHO {[TO] FILE fileid} }
                { [TO] FILE fileid [WITH] ECHO {OFF             } }


                              { pos1           }
.RUN   FILE fileid [ IGNORE { pos1 THRU      } ]
                              { THRU pos2      }
                              { pos1 THRU pos2 }


.SET   var [TO] expression


.SYSTEM 'command'


.TABLE tableref


 UPDATE [dbname.]tblname SET colname = :fldname [... ,colname = :fldname]

        WHERE colname = :fldname [{AND} colname = :fldname]
                                 {OR }
```

DATA DICTIONARY VIEWS

Data Dictionary/Directory view contents are listed alphabetically.
Those views with an 'X' suffix restrict the data returned to rows associated with the executing user.

AccLogRules	UserName,	DatabaseName,	TVMName,	
	AcrAlterFunction,	AcrCheckpoint,	AcrCreateDatabase,	
	AcrCreateFunction,	AcrCreateMacro,	AcrCreateTable,	
	AcrCreateUser,	AcrCreateView,	AcrCreateProcedure,	
	AcrDelete,	AcrDropDatabase,	AcrDropFunction,	
	AcrDropMacro,	AcrDropTable,	AcrDropUser,	
	AcrDropView,	AcrDropProcedure,	AcrDump,	
	AcrExecute,	AcrExecuteFunction,	AcrExecuteProcedure,	
	AcrGrant,	AcrIndex,	AcrInsert,	
	AcrReference,	AcrRestore,	AcrSelect,	
	AcrUpdate,	AcrCreateTrigger,	AcrDropTrigger,	
	AcrCreateRole,	AcrDropRole,	AcrCreateProfile,	
	AcrDropProfile,	AcrAlterProcedure,	CreatorName,	
	CreateTimeStamp			

AccessLog	LogDate,	LogTime,	LogonDate,	
	LogonTime,	LogicalHostId,	IFPNo,	
	SessionNo,	UserName,	AccountName,	
	OwnerName,	AccessType,	Frequency,	
	EventCount,	Result,	DatabaseName,	
	TVMName,	ColumnName,	StatementType,	
			StatementText	

AccountInfo[X]	UserName,	AccountName,	UserOrProfile

AllRights	UserName,	DataBaseName,	TableName,	
	ColumnName,	AccessRight,	GrantAuthority,	
	GrantorName,	AllnessFlag,	CreatorName,	
	CreateTimeStamp			

AllRoleRights	RoleName,	DataBaseName,	TableName,	ColumnName,	
	AccessRight,	GrantorName,	CreateTimeStamp		

AllSpace[X]	Vproc,	DataBaseName,	AccountName,	TableName,	
	MaxPerm,	MaxSpool,	MaxTemp,		
	CurrentPerm,	CurrentSpool,	CurrentTemp,		
	PeakPerm,	PeakSpool,	PeakTemp		

AllTempTables[X]	HostNo,	SessionNo,	UserName,	
	B_DatabaseName,	B_TableName,	E_TableId	

All_RI_Children	IndexID,	IndexName,	ChildDB,	
	ChildTable,	ChildKeyColumn,	ParentDB,	
	ParentTable,	ParentKeyColumn,	InconsistencyFlag,	
	CreatorName,	CreateTimeStamp		

```
                { IndexID,     IndexName,       ParentDB,          }
All_RI_Parents  { ParentTable, ParentKeyColumn, ChildDB,           }
                { ChildTable,  ChildKeyColumn,  InconsistencyFlag, }
                { CreatorName, CreateTimeStamp                     }


AMPusage        { AccountName, UserName,  CpuTime, DiskIO, }
                { Vproc,       VprocType, Model            }


                { DataBaseName, TableName, EventNum,             }
                { Original_DatabaseName,   Original_TableName,   }
Association     { Original_TableKind,      Original_Version,     }
                { Original_ProtectionType, Original_JournalFlag, }
                { Original_CreatorName,    Original_CommentString}


CharSets        { CharSetName }


CharTranslations { CharSetName, CharSetId, InstallFlag, }
                 { E2I,  E2IUp, I2E, I2EUp              }


Children[X]     { Child, Parent }


Collations      { CollName,    CollInstall, CollEqvClass, }
                { CollOrderCS, CollOrderUC                }


                { DataBaseName,   TableName,           ColumnName,                }
                { ColumnFormat,   ColumnTitle,         SSParameterType,           }
                { ColumnType,     ColumnLength,        DefaultValue,    Nullable, }
Columns[X]      { Commentstring,  DecimalTotalDigits,  DecimalFractionalDigit,    }
                { ColumnId,       UppercaseFlag,       Compressible,              }
                { CompressValue,  ColumnConstraint,    ConstraintCount,           }
                { CreatorName,    CreateTimeStamp,     LastAlterName,             }
                { LastAlterTimeStamp,  CharType,       IdColType                  }
                { AccessCount,    LastAccessTimeStamp, CompressValueList          }


                { DatabaseName,       TableName,              ColumnName,   }
ColumnStats     { ColumnType,         ColumnLength,           ColumnFormat, }
                { DecimalTotalDigits, DecimalFractionalDigit, SeqNumber     }


CSPSessionInfo  { SessionNo, HostNo, StartMBox, LogonSource }


                { DataBaseName,       CreatorName,         OwnerName,          }
                { AccountName,        ProtectionType,      JournalFlag,        }
Databases[X]    { PermSpace,          SpoolSpace,          TempSpace,          }
                { CommentString,      CreateTimeStamp,     LastAlterName,      }
                { LastAlterTimestamp, DBKind, AccessCount, LastAccessTimeStamp }


Databases2      { DataBaseName, DataBaseId, UnResolvedRICount }
```

```
DataBase_Default_Journals[X] { DataBaseName, Journal_DB, JournalName }


DBCInfo            { InfoKey, InfoDate }


                   { DatabaseName, AccountString, ExplainFlag,   }
                   { ObjFlag,      SQLFlag,       StepFlag,      }
DBQLRules          { SummaryFlag,  ThresholdFlag, TextSizeLimit, }
                   { SummaryVal1,  SummaryVal2,   SummaryVal3,   }
                   { ThreshValue                                 }


DeleteAccessLog    { LogDate, LogTime }


                   { LogicalHostId,    SessionNumber,          }
                   { CoordTaskId,      RunUnitId,              }
DeleteOldInDoubt   { LogonUserName,    ResolvingUserLogonName, }
                   { CommitOrRollback, UserLogonDate,          }
                   { UserLogonTime,    CompletionDate,         }
                   { CompletionTime,   Options                 }


                   { Vproc,       DataBaseName, AccountName, }
DiskSpace[X]       { MaxPerm,     MaxSpool,     MaxTemp,     }
                   { CurrentPerm, CurrentSpool, CurrentTemp, }
                   { PeakPerm,    PeakSpool,    PeakTemp     }


                   { CreateDate,      CreateTime,         EventNum,   EventType,     }
                   { UserName,        DatabaseName,       ObjectType, AllAMPsFlag,   }
Events[X]          { RestartSeqNum,   OperationInProcess, TableName,  CheckpointName,}
                   { LinkingEventNum, DataSetName,        LockMode,   JournalUsed,   }
                   { JournalSaved,    IndexPresent,       DupeDumpSet                }


                      { CreateDate,     CreateTime,   EventNum,     EventType,}
Events_Configuration[X] { UserName,       LogProcessor, PhyProcessor, Vproc,    }
                      { ProcessorState, RestartSeqNum                         }


                   { CreateDate,  CreateTime,  EventNum,    EventType,      }
Events_Media[X]    { UserName,    DataSetName, VolSerialID, VolSequenceNum, }
                   { DupeDumpSet                                          }


                   { DatabaseName,     FunctionName,   SpecificName,       }
                   { FunctionId,       NumParameters,  ParameterDataTypes, }
Functions          { FunctionType,     ExternalName,   SrcFileLanguage,    }
                   { NoSQLDataAccess,  ParameterStyle, DeterministicOpt,   }
                   { NullCall,         PrepareCount,   ExecProtectionMode  }
                   { ExtFileReference, CharacterType,  Platform            }


                   { TheDate,        TheTime,               Event_Tag           }
                   { Category,       Severity,              Primary_Part_Number,}
                   { Revision_Level, Secondary_Part_Number, Serial_Number,      }
Hardware_Event_Log { PMA,            Module_Type,Slot,      Slot_Type,          }
                   { SubSlot,        SubSlot_Type,          FW_Version,         }
                   { Vcc_Margin,     Frequency_Margin,      Vcc_Volts,          }
                   { Vcc_Amps,       Temperature,           Line,               }
                   { Text,           Error_Data                                 }
```

```
HostInfo           { LogicalHostId, HostName, DefaultCharSet }


                   { DataBaseName,        TableName,      IndexName,      }
IndexConstraints   { IndexNumber,         ConstraintType, ConstraintText, }
                   { ConstraintCollation, CollationName,  CreatorName     }
                   { CreateTimeStamp                                      }


                   { DatabaseName,       TableName,              IndexNumber,    }
IndexStats         { IndexName,          IndexType,              UniqueFlag,     }
                   { ColumnPosition,     ColumnName,             ColumnType,     }
                   { ColumnLength,       ColumnFormat,                           }
                   { DecimalTotalDigits, DecimalFractionalDigit, IndexStatistics }


                   { DataBaseName,    TableName,      IndexNumber,        }
                   { IndexType,       UniqueFlag,     IndexName,          }
Indices[X]         { ColumnName,      ColumnPosition, CreatorName,        }
                   { CreateTimeStamp, LastAlterName,  LastAlterTimestamp, }
                   { IndexMode,       AccessCount,    LastAccessTimeStamp }


                   { LogicalHostId,  SessionNumber,    CoordTaskId,           }
InDoubtLog         { RunUnitId,      LogonUserName,    ResolvingUserLogonName, }
                   { UserLogonDate,  UserLogonTime,    CompletionDate,        }
                   { CompletionTime, CommitOrRollback, Options                }


Journals[X]        { Tables_DB,   TableName,  }
                   { Journals_DB, JournalName }


                   { LogDate,   LogTime,       UserName,   AccountName, }
LogOnOff           { Event,     LogicalHostId, IFPNo,      SessionNo,   }
                   { LogonDate, LogonTime,     LogonSource             }


LogonRules         { UserName,     LogicalHostId, LogonStatus,    }
                   { NullPassword, CreatorName,   CreateTimeStamp }


                   { DatabaseName,       TableName,              StatisticsId,    }
MultiColumnStats   { ColumnPosition,     ColumnName,             ColumnType,      }
                   { ColumnLength,       ColumnFormat,                            }
                   { DecimalTotalDigits, DecimalFractionalDigit, ColumnStatistics }


OwnerDB            { Id, Name, NameI }


                   { ProfileName,      DefaultAccount,   DefaultDB,         }
                   { SpoolSpace,       TempSpace,        ExpirePassword,    }
ProfileInfo[X]     { PasswordMinChar,  PasswordMaxChar,  PasswordDigits,    }
                   { PasswordSpecChar, MaxLogonAttempts, LockedUserExpire,  }
                   { PasswordReuse,    CommentString,    CreatorName,       }
                   { CreateTimeStamp,  LastAlterName,    LastAlterTimeStamp }
```

```
                      { ProcId,         CollectTimeStamp, QueryID,           }
                      { UserID,         AcctString,       ExpandAcctString,  }
QryLog                { SessionID,      LogicalHostID,    RequestNum,        }
                      { LogonDateTime,  AcctStringTime,   AcctStringHour,    }
                      { AcctStringDate, AppID,            ClientID,          }
                      { QueryBrand,     ProfileID,        StartTime,         }
                      { FirstStepTime,  FirstRespTime,    LastRespTime,      }
                      { NumSteps,       NumStepswPar,     MaxStepsInPar,     }
                      { NumResultRows,  ResultRowSize,    TotalIOCount,      }
                      { TotalCPUTime,   ErrorCode,        ErrorText,         }
                      { TDQMFlag,       AbortFlag,  CacheFlag,  QueryText,   }
                      { HotAmp1CPU,     HotAmp2CPU, HotAmp3CPU,              }
                      { LowAmp1CPU,     LowAmp2CPU, LowAmp3CPU, AvgAmpCPUSec,}
                      { HotAmp1IO,      HotAmp2IO,  HotAmp3IO,               }
                      { LowAmp1IO,      LowAmp2IO,  LowAmp3IO,  AvgAmpIOCnt, }
                      { SpoolUsage,     <Extra fields>                       }


QryLogExplain         { ProcId,   CollectTimeStamp, QueryID, }
                      { ExpRowNo, ExplainText                }


                      { ProcId,             CollectTimeStamp, QueryID,          }
QryLogObjects         { ObjectDatabaseName, ObjectTableName,  ObjectColumnName, }
                      { ObjectID,           ObjectNum,        ObjectType,       }
                      { FreqOfUse,          TypeOfUse                           }


QryLogSQL             { ProcId,   CollectTimeStamp, QueryID, }
                      { SqlRowNo, SqlTextInfo                }


                      { ProcId,        CollectTimeStamp, QueryID,                }
                      { StepLev1Num,   StepLev2Num,      StepName,               }
                      { StepStartDate, StepStopDate,     CPUTime,                }
                      { IOCount,       RowCount,                                 }
QryLogSteps           { HotAmp1CPU,    HotAmp2CPU,       HotAmp3CPU,             }
                      { LowAmp1CPU,    LowAmp2CPU,       LowAmp3CPU, AvgAmpCPUSec,}
                      { HotAmp1IO,     HotAmp2IO,        HotAmp3IO,              }
                      { LowAmp1IO,     LowAmp2IO,        LowAmp3IO,  AvgAmpIOCnt, }
                      { <Extra fields>                                           }


QryLogSummary         { ProcId,     CollectTimeStamp, SessionID,          }
                      { QueryCount, QuerySeconds,     LowHist,   HighHist }


RCC_Configuration[X]  { EventNum, LogProcessor,   PhyProcessor, }
                      {  Vproc,   ProcessorState, RestartSeqNum }


RCC_Media[X]          { EventNum, VolSerialId, VolSequenceNum, DupeDumpSet }


                      { TheDate, TheTime, Vproc,           NodeId,          }
                      { Secs,             NCPUs,           GroupId,         }
ResCPUUsageByAMPView  { AMPWorkTaskExec,  AMPWorkTaskServ, AMPMiscUserExec, }
                      { AMPMiscUserServ,  AMPTotalUserExec, AMPTotalUserServ }
```

```
                      { TheDate, TheTime, Vproc,           NodeId,          }
                      { Secs,             NCPUs,           GroupId,         }
                      { PEDispExec,       PEDispServ,      PEParsExec,      }
ResCPUUsageByPEView   { PEParsServ,       PESessExec,      PESessServ,      }
                      { PEMiscUserExec,   PEMiscUserServ,  PETotalUserExec, }
                      { PETotalUserServ                                     }


                      { TheDate, TheTime, Vproc,                NodeId,              }
                      { Secs,             GroupId,              NCPUs,               }
                      { CPUBusy,          CPUOpSys,             CPUWaitIO,           }
                      { DiskSegmentIO,    LogicalDeviceIO,      LogicalDeviceReads,  }
                    { LogicalDeviceWrites, LogicalDeviceReadKB, LogicalDeviceWriteKB,}
                      { MemAgings,        MemBackupCompleteSegs, MemBackupPartialSegs,}
                      { MemFails,         MemFreeKB,            MemSize,             }
ResGeneralInfoView    { MemTextAllocs,    MemVprAllocs,                              }
                      { NetAttempts,      NetBackoffs,                               }
                      { NetChannelSR,     NetMultiIO,           NetPtoPIO,           }
                      { NetReadKB,        NetReads,             NetWriteKB,          }
                      { NetWrites,        PageOrSwapIO,         ProcActiveAvg,       }
                      { ProcBlksDBLock,   ProcBlockedAvg,       ProcBlocks,          }
                      { ProcWaits,        UserStmtsArriving,    UserStmtsInProgress  }


                      { TheDate, TheTime,  NodeId, VprId,     HstId, HstType,  }
                      { Secs,              NominalSecs,       GroupId,         }
                      { CollectIntervals,  HostBlockReads,    HostBlockWrites, }
ResShstGroupView      { HostMessageReads,  HostMessageWrites, HostReadKB,      }
                      { HostWriteKB,       HostQLenSum,       HostQLenMax,     }
                      { HostReadFails,     HostWriteFails                      }


                      { TheDate, TheTime, NodeId, VprId,    LdvId, LdvType,  }
                      { Secs,             NominalSecs,      GroupId,         }
                      { CollectIntervals, LdvConcurrentSum, LdvOutReqSum,    }
ResSldvGroupView      { LdvReads,         LdvWrites,        LdvReadKB,       }
                      { LdvWriteKB,       LdvReadRespTot,   LdvWriteRespTot, }
                      { LdvReadRespMax,   LdvWriteRespMax,  LdvReadRespSq,   }
                      { LdvWriteRespSq,   LdvConcurrentMax, LdvOutReqMax,    }
                      { LdvOutReqTime                                        }


                      { IndexID,     IndexName,      ChildDbID,         }
RI_Child_Tables       { ChildTID,    ChildKeyFID,    ParentDbID,        }
                      { ParentTID,   ParentKeyFID,   InconsistencyFlag, }
                      { CreatorName, CreateTimeStamp                    }


                      { IndexID,           IndexName,   ChildDB,        }
RI_Distinct_Children  { ChildTable,        ParentDB,    ParentTable     }
                      { InconsistencyFlag, CreatorName, CreateTimeStamp }


                      { IndexID,           IndexName,   ParentDB,       }
RI_Distinct_Parents   { ParentTable,       ChildDB,     ChildTable      }
                      { InconsistencyFlag, CreatorName, CreateTimeStamp }


                      { IndexID,     IndexName,      ParentDbID,        }
RI_Parent_Tables      { ParentTID,   ParentKeyFID,   ChildDbID,         }
                      { ChildTID,    ChildKeyFID,   InconsistencyFlag, }
                      { CreatorName, CreateTimeStamp                    }
```

```
RoleInfo[X]           { RoleName,       CreatorName, CommentString, }
                      { CreateTimeStamp                             }

RoleMembers[X]        { RoleName, Grantee,     GranteeKind,           }
                      { Grantor,  WhenGranted, DefaultRole, WithAdmin }

                      { ExpirePassword,   PasswordMinChar,  PasswordMaxChar,  }
SecurityDefaults      { PasswordDigits,   PasswordSpecChar, MaxLogonAttempts, }
                      { LockedUserExpire, PasswordReUse                       }

SecurityLog[X]        { LogDate,     LogTime,      LogType,   UserName, }
                      { AccountName, DataBaseName, TableName, Text      }

                      { UserName,         AccountName,     SessionNo,        }
                      { DefaultDataBase,  IFPNo,           Partition,        }
SessionInfo[X]        { LogicalHostId,    HostNo,          CurrentCollation, }
                      { LogonDate,        LogonTime,       LogonSequenceNo,  }
                      { LogonSource,      ExpiredPassword, TwoPCMode,        }
                      { Transaction_Mode, CurrentRole,     LogonAcct         }

ShowColChecks         { DatabaseName, TableName,   ColumnName, }
                      { ColCheck,     CreatorName, CreateTimeStamp

ShowTblChecks         { DatabaseName, TableName,   CheckName,      }
                      { TblCheck,     CreatorName, CreateTimeStamp }

                      { TheDate,    TheTime,        Event_Tag,   }
                      { Category,   Severity,       PMA,         }
Software_Event_Log    { Vproc,      Partition,Task, TheFunction, }
                      { SW_Version  Line,           Text,        }
                      { StackTrace, Error_Data                   }

                      { DataBaseName,         TableName,          Version,             }
                      { TableKind,            ProtectionType,     JournalFlag,         }
Tables[X]             { CreatorName,          RequestText,        CommentString,       }
                      { ParentCount,          ChildCount,         NamedTblCheckCount,  }
                      { UnnamedTblCheckExist, PrimaryKeyIndexId,  CreateTimeStamp,     }
                      { LastAlterName         LastAlterTimeStamp, RequestTextOverFlow, }
                      { AccessCount,          LastAccessTimeStamp                      }

Tables2               { TVMName,     TVMId,     DatabaseId, }
                      { ParentCount, ChildCount             }

TableSize[X]          { Vproc,     DataBaseName, AccountName, }
                      { TableName, CurrentPerm,  PeakPerm     }

TableText[X]          { DataBaseName, TableName, TableKind, }
                      { RequestText,  LineNo                }
```

```
Table_LevelConstraints  { DatabaseName,   TableName,   ConstraintName, }
                        { ConstraintText, CreatorName, CreateTimeStamp }


                  { DataBaseName,  TableName,          TriggerName,    }
                  { EnabledfFlag,  ActionTime,         Event,          }
Triggers          { Kind,          OrderNumber,        TriggerComment  }
                  { RequestText,   CreatorName,        CreateTimeStamp,}
                  { LastAlterName, LastAlterTimeStamp, AccessCount,    }
                  { LastAccessTimeStamp                                }


UserDB            { Id, Name }


                  { DataBaseName, TableName,   ColumnName,     }
UserGrantedRights { Grantee,      AccessRight, GrantAuthority, }
                  { AllnessFlag,  CreatorName, CreateTimeStamp }


                  { DataBaseName, TableName,      ColumnName, }
UserRights        { AccessRight,  GrantAuthority, GrantorName }
                  { CreatorName,  CreateTimeStamp             }


UserRoleRights    { RoleName,    DataBaseName, TableName,      ColumnName, }
                  { AccessRight, GrantorName,  CreateTimeStamp             }


                  { UserName,            CreatorName,     PasswordLastModDate, }
                  { PasswordLastModTime, Ownername,       PermSpace            }
                  { SpoolSpace,          TempSpace,       ProtectionType,      }
                  { JournalFlag,         StartUpString,   DefaultAccount,      }
Users             { DefaultDataBase,     CommentString,   DefaultCollation     }
                  { PasswordChgDate,     LockedDate,      LockedTime,          }
                  { LockedCount,         TimeZoneHour,    TimeZoneMinute,      }
                  { DefaultDateForm,     CreateTimeStamp, LastAlterName,       }
                  { LastAlterTimeStamp,  DefaultCharType, RoleName,            }
                  { ProfileName,         AccessCount,     LastAccessTimeStamp  }


User_Default_Journals[X]  { UserName, Journal_DB, JournalName }
```

Builtin Values and Functions

Built in Value	Value Returned	Data Type
ACCOUNT	Current User Account	VARCHAR(30)
CURRENT_DATE	Current Date	DATE
CURRENT_TIME [(n)]	Current Time	TIME(n) WITH TIMEZONE
CURRENT_TIMESTAMP [(n)]	Current Date/Time	TIMESTAMP(n) WITH TIMEZONE
DATABASE	Current Default DB	VARCHAR(30)
DATE	Current Date	DATE
NULL	The NULL value	As required
PARTITION	The Index Partition	INTEGER
ROWID	The Internal Row Id	INTEGER
SESSION	Current Session Number	INTEGER
TIME	Current Time	FLOAT
USER	Current User Id	VARCHAR(30)

Descriptor Functions	Value Returned	Data Type
BYTE[S] (arg)	Length of arg	INTEGER
CHAR[ACTERS] (string)	Length of str	INTEGER
MCHAR[ACTERS] (string)	Length (Mbyte Chars)	INTEGER
CHAR[ACTER]_LENGTH(string)	Length of str (Chars)	INTEGER
OCTET_LENGTH (string [charset])	Length of str (Bytes)	INTEGER
FORMAT (arg)	Format of arg	VARCHAR(30)
NAMED (arg)	Name assigned to arg	VARCHAR(30)
TITLE (arg)	Title on arg	VARCHAR(60)
TYPE (arg)	Data Type of arg	VARCHAR(??)

Math Functions	Value Returned	Data Type
ABS (num)	Absolute value	Same as arg
ACOS (num)	ArcCosine	FLOAT
ACOSH (num)	Hyperbolic ArcCosine	FLOAT
ASIN (num)	ArcSine	FLOAT
ASINH (num)	Hyperbolic ArcSine	FLOAT
ATAN (num)	ArcTangent	FLOAT
ATANH (num)	Hyperbolic ArcTangent	FLOAT
COS (num)	Cosine	FLOAT
COSH (num)	Hyperbolic Cosine	FLOAT
EXP (num)	e to the power arg	FLOAT
LOG (num)	Base 10 Logarithm	FLOAT
LN (num)	Base e Logarithm	FLOAT
SIN (num)	Sine	FLOAT
SINH (num)	Hyperbolic Sine	FLOAT
SQRT (num)	Square Root	FLOAT
TAN (num)	Tangent	FLOAT
TANH (num)	Hyperbolic Tangent	FLOAT

Builtin Values and Functions - Continued

Conversion Functions		Value Returned	Data Type
ADD_MONTHS	(date, n) (timestamp, n)	Add 'n' months to date	DATE TIMESTAMP
CASE	WHEN exp1 THEN val1 [WHEN exp2 THEN val2] ... [ELSE valn] END	Value substitution	Same as valn
CASE_N	(expr [.. ,expr] [,NO CASE [OR UNKNOWN]] [,UNKNOWN])		INTEGER
CAST	(expr AS typeinfo)	Type/Format conversion	as in typeinfo
CHAR2HEXINT	(string)	HEX display of string	VARCHAR
COALESCE	(expr1, expr2 [.. ,exprn])	First non-Null expr	Same as expr
DATE	'YYYY-MM-DD'	YYYY-MM-DD as a date	DATE
EXTRACT	(part FROM date)	Day, Hour, Minute etc	'part' type
HASHROW	(expr, expr)	The Row Hash	
HASHBUCKET	(expr)	The Hash Bucket	
HASHAMP	(expr)	The Primary AMP	
HASHBACKAMP	(expr)	The Backup AMP	
INDEX	(string, substr)	Start pos. of substr	INTEGER
LOWER	(string)	String in LowerCase	CHAR
MINDEX	(MBstring, substr)	Start pos. of substr	INTEGER
NULLIF	(expr1, expr2)	Null if e1=e2 else e1	Same as expr1
NULLIFZERO	(num)	NULL if arg is Zero	Same as arg
(start,end)	OVERLAPS (start,end)	Dates/intervals overlap?	BOOLEAN
POSITION	(str1 IN str2)	Start pos of str1 in str2	INTEGER
RANGE_N	(expr BETWEEN start [AND end] [EACH size] [...,] [,NOT IN RANGE [OR UNKNOWN]] [,UNKNOWN])		INTEGER
RANDOM	(low-bound, high_bound)	A random number	FLOAT
SOUNDEX	( )		
SUBSTR	(string, start, len)	Sub-String of string	CHAR
TRIM	([[BOTH] [LEADING] [chr] FROM] string) [TRAILING]	Remove blanks (or chr)	CHAR
UPPER	(string)	String in UpperCase	CHAR
WIDTH_BUCKET	( )		
ZEROIFNULL	(num)	Zero if arg is NULL	Same as arg

Builtin Values and Functions - Continued

Aggregate and OLAP Functions		Description
AVG	([DISTINCT] arg)	Average value
CORREL	(y, x)	Correlation
COUNT	([DISTINCT] arg)	Number occurrences
COUNT	(*)	Number of rows
COVARIANCE	(y, x)	Covariance
COVAR_SAMP	(y, x)	Sample Covariance
CSUM	(col, sort-expr [… ,sort-expr])	Cumulative Sum
GCOUNT	(col)	?
GSUM	(col)	?
KURTOSIS	(arg)	Kurtosis
LINREGSLOPE	(y, x)	Slope: Linear Reg
LINREGINTERCEPT	(y, x)	Intercept: Linear Reg
MAX[IMUM]	(arg)	Maximum value
MIN[IMUM]	(arg)	Minimum value
MAVG	(col, #rows, sort-expr [… ,sort-expr])	Moving Average
MDIFF	(col, #rows, sort-expr [… ,sort-expr])	Moving Difference
MLINREG	(col, #rows, sort-expr)	Linear Regression
MSUM	(col, #rows, sort-expr [… ,sort-expr])	Moving Sum
PERCENT_RANK	() OVER ([PARTITION BY spec] ORDER BY spec [ASC | DESC])	
QUANTILE	(#partitions, sort-expr [… ,sort-expr])	Quantile position
RANK	(sort-expr [… ,sort-expr])	Rank Position
RANK	() OVER ([PARTITION BY spec] ORDER BY spec [ASC | DESC])	
REGR_AVGX	(y, x)	Avg of x values in regression
REGR_AVGY	(y, x)	Avg of y values in regression
REGR_COUNT	(y, x)	# non-null pairs in regression
REGR_R2	(y, x)	R squared of Regression
REGR_SXX	(y, x)	? in regression
REGR_SXY	(y, x)	? in regression
REGR_SYY	(y, x)	? in regression
SKEW	([DISTINCT] arg)	Skew
STDEV	(arg)	Standard Deviation
STDEVP	(arg)	Standard Deviation (pop)
STDDEV_POP	([DISTINCT] arg)	Population Std Deviation
STDDEV_SAMP	([DISTINCT] arg)	Sample Std Deviation
SUM	([DISTINCT] arg)	Sum of values
SUM	(arg) OVER ([PARTITION BY spec] [ORDER BY spec] ROWS {windowsize} PRECEDING [ASC | DESC]) {UNBOUNDED }	
VARIANCE	(arg)	Variance
VARIANCEP	(arg)	Population Variance
VAR_POP	([DISTINCT] arg)	Population Variance
VAR_SAMP	([DISTINCT] arg)	Sample Variance

Maximum Limits on the Teradata RDBMS

SYSTEM	Number of Data bases		4.2 Billion	
	Message length		1 MB	
	SQL request length		1 MB	
	Active transactions		2,048	
	Data Parcel Length		65,104	
	Parcels in one message		256	
	SQL title length		60	
	String constant length		255	
	Data Format Descriptor Length		30	
	Error message text in failure parcel		255	
	Sessions per gateway (Max 1 gateway / Node)		1,200	
	Sessions per PE		120	
	Concurrent Utility jobs		15	
	Vprocs per system		16,384	
	Vprocs per Node		128	
	Data capacity per AMP - Unformatted		1.3	Tbyte
DATABASE	Tables per database		32,000	
	Journal tables per database		1	
	Columns per table		2,048	
	LOB columns per table		32	
	Columns per View / Spool file		512	
	Block Size		130,560	
	Row size	(approx)	64,256	
	Column size	(approx)	64,000	
	LOB size		2 GB	
	Column / Table name length		30	
	Number of fields per index		62	
	Secondary / Join indexes per table		32	
	Table level constraints per table		100	
	Referential constraints per table		64	
	Tables that can reference a table		64	
	Columns in Foreign & Parent key		16	
	View / Macro nesting levels		8	
	Rows per table	limited by space available		
SESSION	Spool Files		2048	
	Global Temporary Tables		1000	
	Volatile Temporary Tables		2000	
	Parallel steps performed (If no channels)		20	
	Number of channels		10	

(Redistribution across AMPs uses 4 channels,
Non prime Index (without redistribution) uses 2 channels)

TERADATA DOCUMENTATION

Document Name	Manual #
Release Summary for Version 2 Release 5	BD35-1098
Introduction to the Teradata RDBMS	BD35-1091
Messages Reference	BD35-1096
Data Dictionary Reference	BD35-1092
Database Administration Guide	BD35-1093
Database Design Guide	BD35-1094
Database Window Reference	BD35-1095
Performance Optimization	BD35-1097
Resource Usage Macros and Tables	BD35-1099
Security Administration Guide	BD35-1100
Database Utilities Reference	BD35-1102
SystemFE Macros	BD35-1103
Teradata SQL Reference	BD35-1101
Performance Monitor (PM/API) Reference	BD35-1090
SQL Mapping and Collation Tables	BD35-1105
International Character Set Support	BD35-1125
SQL / Data Dictionary Quick Reference	BD35-1510
Utilities Quick Reference	BD35-1511
Teradata Tools and Utilities (TTU) 7.0 Release Summary	BD35-2427
Multiload Reference	BD35-2409
Fast Export Reference	BD35-2410
Fastload Reference	BD35-2411
Archive/Recovery Reference Manual for Channel attached	BD35-2412
BTEQ Reference	BD35-2414
Tpump Reference	BD35-3021
Teradata Manager Installation Guide	BD35-2402
Getting Started with Teradata Manager	BD35-2428
SQL Assistant Users Guide	BD35-2430
TeraBuilder Operators Reference	BD35-2433
TeraBuilder Operator Programming Reference	BD35-2435
TeraBuilder Reference	BD35-2436
TDQM Administrators Guide	BD35-3027
TDQM Users Guide	BD35-3028
TDQM Programmers Guide	BD35-3029
MDS Programmers Guide	BD35-3037
MDS Installation and Configuration Guide	BD35-3045
MDS Administrators Guide	BD35-3117

TERADATA DOCUMENTATION - Continued

Document Name	Manual #
ODBC driver for Windows User Guide	BD35-3061
JDBC driver Installation and User Guide	BD35-2403
CLI2 Developers Kit for Windows	BD35-2408
Call Level Interface V2 for Channel attached	BD35-2417
Call Level Interface V2 for Network attached	BD35-2418
Access Module Programming Reference	BD35-2424
Access Module Reference	BD35-2425
TDP Reference	BD35-2416
Teradata Application programming	BD35-2446
Data Definition Language Processor Reference	BD35-2449
CICS Interface to the Teradata DBS	BD35-2448
IMS Interface to the Teradata DBS	BD35-2447
TS/API User's Guide	BD35-2419

TDP COMMAND SUMMARY

TDP commands are listed alphabetically. Capital letters are used to indicate the minimum abbreviation required for that keyword.

These commands may be entered from the MVS or VM console, or by a VM/TSO user (Through SMSG or DBCCMD) who has been so AUTHORIZED.

```
          {XMSCELLS}
ADD       {         } SIZe cellsize NUMber numcells
          {CELLS    }


ATTach    ifpname


                      {RESOLVE }
          {userid}    {None    }
AUthoriz  {job   }    {Display }
          {ALL   }    {Any     }
                      {AUthoriz}


        { ALL             }
COMMIT  {                 } COORD name [ HOST id ]
        { SESSION number  }


DETach    ifpname


DISAble   IRF


DISAble   LGUX


DISAble   LOGONS


DISAble   POOL        { ID poolid }
                      { ALL       }


DISable   SESSRSRV


DISAble   SMF         { SUBn  [... SUBn] }
                      { ALL              }


DISAble   TEST
```

```
DISAble  TIMe


DISAble  TMON


DISAble  UAX


DISAble  USEC


Display  CELls [VERify]


Display  { IFP     } [STATE]
         { ifpname }


                 { SESSIONS COORD name [RESOLVED] }
Display  INDoubt {                                } [HOST id]
                 { COORDS                         }


Display  POOL {ID poolid}
              {ALL      }


Display  Queues


         {          { sessnumber } }
         { SESsions { ENDing     } }
Display  {          { ALL        } }
         {                         }
         { JOB jobname             }


Display  SMF


Display  [TDP]


ENAble   IRF


ENAble   LGUX


ENAble   LOGONS
```

```
ENAble   POOL    { ID poolid }
                  { ALL       }


ENAble   SESSRSRV


ENAble   SMF     { SUBn  [... SUBn] }
                 { ALL              }


ENAble   TEST


ENAble   TIMe


ENAble   TMON


ENAble   UAX


ENAble   USEC


         {          { sessnumber } }
         { SESsions {            } }
LOGOFF   {          { ALL        } }
         {                        }
         { JOB jobname            }


LOGOFF   POOL   {ID poolid}
                {ALL      }


MODIFY   POOL   ID poolid   NUM number


         { ALL            }
ROLLBACK {                } COORD name [ HOST id ]
         { SESSION number }


RUN      [TDPn]


                 { comchar }
SET      COMchar {         }
                 { OFF     }


SET      MAXSess numberofsessions


         { CANCEL  }
```

```
SHUTDOWN { QUIck   }
         { Orderly }


STArt    ifpname


STArt    POOL  DDNAME filename


STArt    POOL  NUM number [JOB jobname] [RUN string] [ID poolid]

               CHarset charactersetname

               { LOG username, password [, 'acctid'] }
               { NULLpwd LOG username   [,,'acctid'] }


STOp     ifpname


STOp     POOL    {ID poolid}
                 {ALL      }
```

CONSOLE OPERATOR COMMANDS

```
                 { hostid:session# }
                 { hostid.username }
ABORT    SESSION {      *.username } [LOGOFF] [LIST] [OVERRIDE]
                 { hostid.*        }
                 {      *.*        }
```

```
CNSGET
```

```
        { DBWTIMEOUT }
CNSSET  { LINES      } n
        { STATEPOL   }
        { TIMEOUT    }
```

```
DISABLE   LOGONS
```

```
ENABLE    LOGONS
```

```
        { CONFIG                  }
        { LOGTABLE { tblname }    }
        {          { ALL     }    }
GET     { PERMISSIONS userid@host }
        { RESOURCE                }
        { TIME                    }
        { VERSION                 }
        { SSO                     }
```

← RSS Table names:

SVPR SPMA SCTL IPMA SLDV SHST

```
GRANT     userid@host  priv  [... priv]
```

← all abort grant logons set start log restart interactive(<pgmname>)

```
LOG       ErrorLogText
```

```
QUERY     STATE
```

```
               { NODUMP     }      { COLD     }
RESTART  TPA [ { DUMP={YES} } ]  [ { COLDWAIT } ]  commentstring
               {      {NO } }
```

```
REVOKE    userid@host  priv  [... priv]
```

```
SET       LOGTABLE  { ALL     } {ON }
                    { tblname } {OFF}
```

← See above for table names

```
SET       RESOURCE COLL[ECTION] n1 { NODE  } LOG[GING] n2
                                   { VPROC }
```

```
SET       SESSION COLL[ECTION] n


              { OFF  }
SET       SSO { ON   }
              { ONLY }


                                          { ABORTHOST      }
                                          { CHECKTABLE     }
                                          { CONFIG         }
                                          { DBSCONTROL     }
                                          { DIP            }
                                          { DUMPLOCKLOG    }
                                          { FERRET         }
                                          { FILER          }
                                          { LOCKDISP       }
          [ 1 ]           [-Vn     ]      { QRYCONFIG      }
START     [ 2 ] [,DEBUG] [,VPROC=n]      { QRYSESSN       }
          [ 3 ]           [-V=n    ]      { RCVMANAGER     }
          [ 4 ]                           { REBUILD        }
                                          { RECONFIG       }
                                          { SHOWLOCKS      }
                                          { SYSINIT        }
                                          { TPCCONS        }
                                          { UPDATEDBC      }
                                          { UPDATESPACE    }
                                          { VPROCMANAGER   }
                                          { XGTWGLOBAL -nw }


          { 1 }
STOP      { 2 }
          { 3 }
          { 4 }
```

Subcommand for CHECKTABLE:

```
                              [                { INDEX ID = nnn     } ]
        { ALL TABLES        } [       { ONLY } { UNIQUE INDEXES     } ]
CHECK   { dbname            } [ BUT   {      } { NONUNIQUE INDEXES } ]
        { dbname.tblname    } [       { NOT  } { DATA               } ]
        { <wildcard spec>]} [                { REFERENCE indexes   } ]
                              [                { REFERENCE ID= nnn   } ]

                  {  ONE  }
        AT LEVEL  {  TWO  } [ WITH {NO ERROR LIMIT } ]
                  { THREE } [      {ERROR LIMIT = n} ]

        [ SKIPLOCKS ] [ IN { SERIAL    } ] [ PRIORITY = pg ]
                           { PARRALLEL }
```

Subcommands for FERRET:

```
DATE


DEFRAG[MENT] [/Y]


           [ TO   { file   } ]
ERRORS     [ INTO { STDERR } ]
           [ OVER { ME     } ]
```

```
EXIT                                (Same as END, QUIT, STOP)


              [ keyword ]
H[ELP]   [/L] [ ALL     ]
              [ ?       ]


IN[PUT]  FROM file


         [ TO   { file   } ]
OUT[PUT] [ INTO { STDERR } ]
         [ OVER { ME     } ]


PACKDISK [/Y] [ { FREE [SPACEPERCENT] } [=] n ]
              [ { FSP                 }       ]


Q[UIT]                              (Same as END, EXIT, STOP)


RADIX    [ IN[PUT]  ] [ H[EX] ]
         [ OUT[PUT] ] [ D[EC] ]


         [/S] [ DB        ]
SCANDISK [/M] [ CI        ] [ FIX ]
         [/L] [ MI        ]
              [ FREE[CIS] ]


                  {ALL                                    }
                  {PER[MANENT]                            }
                  {PJ                                     }
                  {SP[OOL]                                }
                  {( PER[MANENT] [, PJ] [, SP[OOL]] )     }
                  {CY[LINDER] {ALL                      } }
SCOPE    [ CLASS {          {drive cyl [... drive cyl]} } ]
                  {TA[BLE]    {ALL                   }    }
                  {           {tableid   [... tableid]}   }
                  {VPROC      {ALL                }       }
                  {           {vprocid}           }       }
                  {           {(vprocid TO vprocid)}      }


           [/S]
SHOWBLOCKS [/M]
           [/L]


SHOWD[EFAULTS]


SHOWFSP


SHOWSPACE [/S]
          [/L]
```

```
    TABLEID   { "dbname.tblname" }
              { 'dbname.tblname' }


    TIME
```

Subcommands for RCVMANAGER:

```
         { STATUS }
    LIST {        } ;
         { LOCKS  }
```

Subcommands for REBUILD:

```
    QUERY


                     {ALL TABLES  } {ALL     }
    REBUILD AMP nnnn {dbname      } {PRIMARY } DATA [, Options] ;
                     {dbname.tname} {FALLBACK}

    REBUILD AMP nnnn FALLBACK TABLES [, Options] ;

    RESTART REBUILD ON AMP nnnn ;
```

Options:

```
        LOG INTO logDB.logTbl

        NO LOCK [ON NO FALLBACK TABLES]

             {DATABASE}
        WITH {TABLE   } LOCK
             {ROWRANGE}
```