```
dml_select ::=
   select_set [ORDER BY order_list]

select_set ::=
   {select_atom | (select_set)}
    [{UNION [ALL] | INTERSECT | MINUS} select_set]

select_atom ::=
   [WITH with_list]
   SELECT [DISTINCT] {* | column_list}
   FROM table_expression
   [WHERE column_arithmetic]
   [START WITH column_arithmetic]
   [CONNECT BY column_arithmetic]
   [GROUP BY {group_list_list | ()}]
   [HAVING column_arithmetic]

dml_insert ::=
   INSERT INTO {tableName [(raw_column_list)]
   | (dml_select [WITH CHECK OPTION])}
   [VALUES (value_list) | dml_select]

dml_update ::=
   UPDATE tableName SET set_list
   [WHERE column_arithmetic]

dml_delete ::=
   DELETE [FROM] tableName
   [WHERE column_arithmetic]

ddl_create_table ::=
   CREATE TABLE tableName
   {(column_type_list) | AS (dml_select)}

ddl_alter_table ::=
   ALTER TABLE tableName
   {{ADD | MODIFY} (column_type_list)
   | ADD table_constraint
   | DROP {column | UNUSED COLUMNS} [CASCADE CONSTRAINTS]
   | DROP {CONSTRAINT constraintName | PRIMARY KEY
             | UNIQUE (raw_column_list)} [CASCADE]
   | SET UNUSED column}

ddl_drop ::=
   DROP {TABLE | VIEW | SEQUENCE | INDEX} objectName

ddl_rename ::=
   RENAME objectName TO objectName

ddl_truncate_table ::=
   TRUNCATE TABLE tableName

ddl_create_view ::=
   CREATE [OR REPLACE] [FORCE | NOFORCE] VIEW viewName
[(raw_column_list)]
   AS dml_select
   [WITH CHECK OPTION [CONSTRAINT constraintName]]
   [WITH READ ONLY [CONSTRAINT constraintName]]

ddl_create_sequence ::=
   CREATE SEQUENCE sequenceName
   [INCREMENT BY value_arithmetic]
   [START WITH value_arithmetic]
   [{MAXVALUE value_arithmetic | NOMAXVALUE}]
   [{MINVALUE value_arithmetic | NOMINVALUE}]
   [{CYCLE | NOCYCLE}]
   [{CACHE value_arithmetic | NOCACHE}]

ddl_alter_sequence ::=
   ALTER SEQUENCE sequenceName
   [INCREMENT BY value_arithmetic]
   [{MAXVALUE value_arithmetic | NOMAXVALUE}]
   [{MINVALUE value_arithmetic | NOMINVALUE}]
   [{CYCLE | NOCYCLE}]
   [{CACHE value_arithmetic | NOCACHE}]

ddl_create_index ::=
   CREATE INDEX indexName ON tableName (list)

dcl_user ::=
   {CREATE | ALTER} USER userName IDENTIFIED BY passWord

dcl_grant ::=
   GRANT privilege_list TO user_list [WITH GRANT OPTION]

dcl_create_role ::=
   CREATE ROLE roleName

dcl_revoke ::=
   REVOKE [ALL | object_privilege_list] ON object
   FROM user_list [CASCADE CONSTRAINTS]

with_list              ::= tableAlias AS (dml_select)
                                     [, with_list]

column_list            ::= column_expression
                                     [, column_list]

column_expression      ::= column_arithmetic
                                     [AS columnAlias]
```

```
column_arithmetic   ::= [NOT] [+ | -] {column | ROWNUM | LEVEL
                              | (column_arithmetic)}
                              [IS [NOT] NULL
                              | binary_operator column_arithmetic
                              | binary_operator [ANY | ALL] (dml_select)
                              | [NOT] { BETWEEN column_arithmetic
                                                  AND column_arithmetic
                                        | IN ({list | dml_select})
                                        | EXISTS (dml_select)}]

column              ::= [PRIOR] {columnAlias | [tableAlias.]columnName [(+)]}
                              | (list) | sequenceName.{CURRVAL | NEXTVAL}
                              | constantValue | singleRowFunction(list)
                              | aggregateFunction
                                         ([DISTINCT | ALL] column_arithmetic)
                              | COUNT(*)
                              | CASE column_arithmetic case_body END

list                ::= column_arithmetic [, list]

case_body           ::= WHEN column_arithmetic
                              THEN column_arithmetic
                              [case_body | ELSE column_arithmetic]

binary_operator     ::= + | - | * | / | || | = | <> | < | > | <= | >=
                              | [NOT] LIKE | AND | OR


table_expression    ::= table [join_expression] [, table_expression]

join_expression     ::= {CROSS | NATURAL} JOIN table
                              | JOIN table USING (raw_column_list)
                              | [LEFT | RIGHT | FULL] [OUTER] JOIN
                                        ON column_arithmetic
                               [join_expression]

table               ::= {tableName | (dml_select)} [tableAlias]

raw_column_list     ::= columnName [, raw_column_list]

group_list_list     ::= [ROLLUP | CUBE | GROUPING SETS]
                              {group_list | (group_list_list)} [, group_list_list]

group_list          ::= {[tableAlias.]columnName | group_list_list} [, group_list]

order_list          ::= column_arithmetic [ASC | DESC] [, order_list]

value_list          ::= {value_arithmetic | DEFAULT} [, value_list]

value_arithmetic    ::= [NOT] [+|-] {value | (dml_select) | (value_arithmetic)}
                               [binary_operator value_arithmetic]

value               ::= constantValue | singleRowFunction(raw_value_list)

raw_value_list      ::= value_arithmetic [, raw_value_list]

set_list            ::= columnName = {column_arithmetic | DEFAULT} [, set_list]

column_type_list    ::= columnName dataType [DEFAULT value_arithmetic]
                              [column_constraint] [NOT NULL]
                              [, {column_type_list | constraint_list}]

column_constraint   ::= [CONSTRAINT constraintName] column_constraint_type

column_constraint_type        ::= UNIQUE | PRIMARY KEY | references
                                        | CHECK (column_arithmetic)

references          ::= tableName(raw_column_list)

constraint_list     ::= table_constraint [, constraint_list]
                               [ON DELETE {CASCADE | SET NULL}]

table_constraint    ::= [CONSTRAINT constraintName] table_constraint_type

table_constraint_type         ::= {{UNIQUE | PRIMARY KEY} (raw_column_list)
                                        | FOREIGN KEY (raw_column_list) references}
                                        | CHECK (column_arithmetic)

privilege_list                ::= privilege [, privilege_list]

privilege           ::= {CREATE | DROP}
                              {USER | [ANY] {TABLE | VIEW | SEQUENCE | PROCEDURE}}
                              | CREATE SESSION
                              | BACKUP [ANY] TABLE
                              | object_privilege ON objectName
                              | roleName

object_privilege    ::= EXECUTE | {ALTER | DELETE | INDEX | INSERT
                              | REFERENCES | SELECT | UPDATE} [(raw_column_list)]

user_list           ::= {userName | roleName | PUBLIC} [, user_list]

object_privilege_list         ::= object_privilege [, object_privilege_list]
```

Avtor: Janoš Vidali, 2006

Avtor ne odgovarja za napake v besedilu.