CS 210 Spring 2012
Project Grammar
Version 1

statement ::= { admin_statement; | ddl_statement; | dml_statement; | query_statement; }

admin_statement ::= { exit | print | read | backup }
print ::= { PRINT DICTIONARY | PRINT table_name }
exit ::= EXIT
read ::= READ file_name
backup::= BACKUP TO file_name
restore ::= RESTORE FROM file_name
ddl_statement ::= { define_table | rename | drop | define_index }
define_table ::= DEFINE TABLE table_name HAVING FIELDS (extended_field_list)
rename ::= RENAME TABLE table_name TO table_name
drop ::= DROP TABLE table_name
define_index ::= DEFINE INDEX ON table_name (field_name)
dml_statement ::= { delete | insert | update }
delete ::= DELETE table_name [WHERE boolean_expression]
insert ::= INSERT (value_list) INTO table_name
update ::= UPDATE table_name SET field_name = value [WHERE
boolean_expression]
query_statement ::= { selection | projection | join | intersection | union | minus }
selection ::= SELECT query_list [WHERE boolean_expression]
projection ::= PROJECT query_list OVER field_list
join ::= JOIN query_list AND query_list
intersection ::= INTERSECT query_list AND query_list
union ::= UNION query_list AND query_list
minus ::= MINUS query_list AND query_list

table_name ::= [special_char] letter [title_string]
title_string ::= { letter | digit | special_char } [title_string]
extended_field_list ::= field_name type [, extended_field_list]
field_name ::= letter [field_name]
type ::= { INTEGER | DATE | REAL | VARCHAR | CHAR(integer) | BOOLEAN }
field_list ::= field_name [, field_list]
query_list ::= { table_name | (query_statement) }
boolean_expression ::= field_name relop value
value ::= { integer | real | date | string_expression | boolean }
value_list ::= value[, value_list]
letter ::= { a – z | A – Z }
digit ::= { 0 – 9 }
special_char ::= { _ | $ }
char ::= <any letter>

integer ::= digit[integer]
real ::= { integer[.][integer] | .integer}
date ::= 'dd/dd/dddd'
d ::= digit
string ::= char[string]
string_expression ::= 'string'
boolean ::= {TRUE | FALSE }
relop ::= { = | != | < | > | <= | >= }
file_name ::= 'string' <The string must represent a file address, either relative or absolute>

COMMENTS:

1. Symbols
   a) ::= definition
   b) [] optional
   c) | or
   d) {} defines groups of 'or' clauses
   e) <> A descriptive statement
   f) All other symbols are part of the grammar
2. Capitalization
   a) Lower case words are defined terms in the grammar
   b) Upper case words are reserved words in the grammar
   c) Note that reserved words are NOT case sensitive (e.g., TRUE is equivalent to true)
3. Word spaces
   a) These include space (' '), \t, \n, \r, \f
   b) Punctuations (including ',' ';' '(' ')', or relop) do not require a word space before or after
   c) Otherwise, all reserved words and other grammar entities (including table_name, file_name, field_name, value) must be separated by at least one word space
4. Dates and string_expressions
   a) String_expressions include varchar and char values. The general rule is that anything that might be case sensitive is surrounded by single quotes. This includes string_expressions, file_names, and dates.
   b) Dates are semantically expressed as month/day/year. For example, '01/11/2011' is acceptable, but '1/11/2011' and '01/11/11' are not.
   c) String_expressions and file_names may include spaces but no other word space characters
   d) String_expressions and file_names are the exception to the case insensitive rule: these ARE case sensitive
   e) Both dates and string_expressions must be separated from other entities by a word space (unless the entity is a punctuation)
   f) Char expressions must be followed by an integer in parentheses – the integer describes the mandatory length of the char expression