By following these steps, one can compile and run the lexer and parser for the specified input query.

1. Compile the Yacc File (`saneql_yacc.y`):

```
yacc -d saneql_yacc.y
```

saneql_yacc.y is the yacc grammar file containing parsing rules.

2. Compile the Lex File (`saneql_lex.l`):

```
  lex saneql_lex.l
```

saneql_lex.l is the lex file containing rules to recognize tokens.

3. Compile and Link Lexer and Parser:

```
  gcc y.tab.c lex.yy.c -ll
```

y.tab.c is the C file generated by yacc containing the parser code and lex.yy.c is the C file generated by lex containing the lexer code.

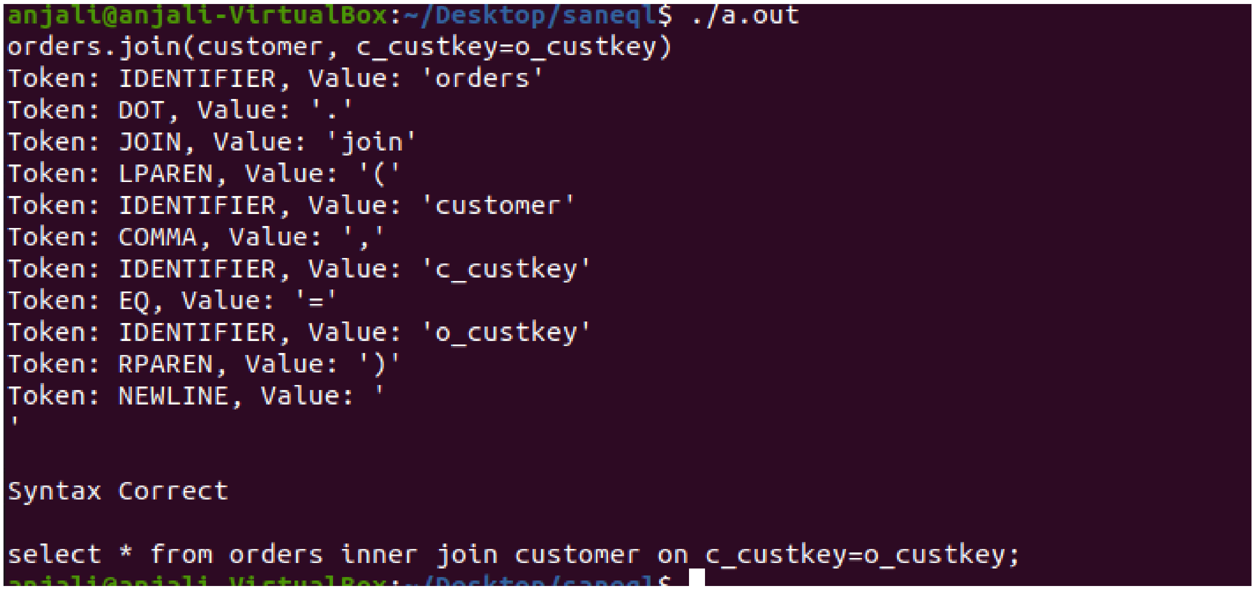
4. Run the Executable:

```
  ./a.out
```

- ./a.out executes the compiled program.

Sample Queries

Join Query

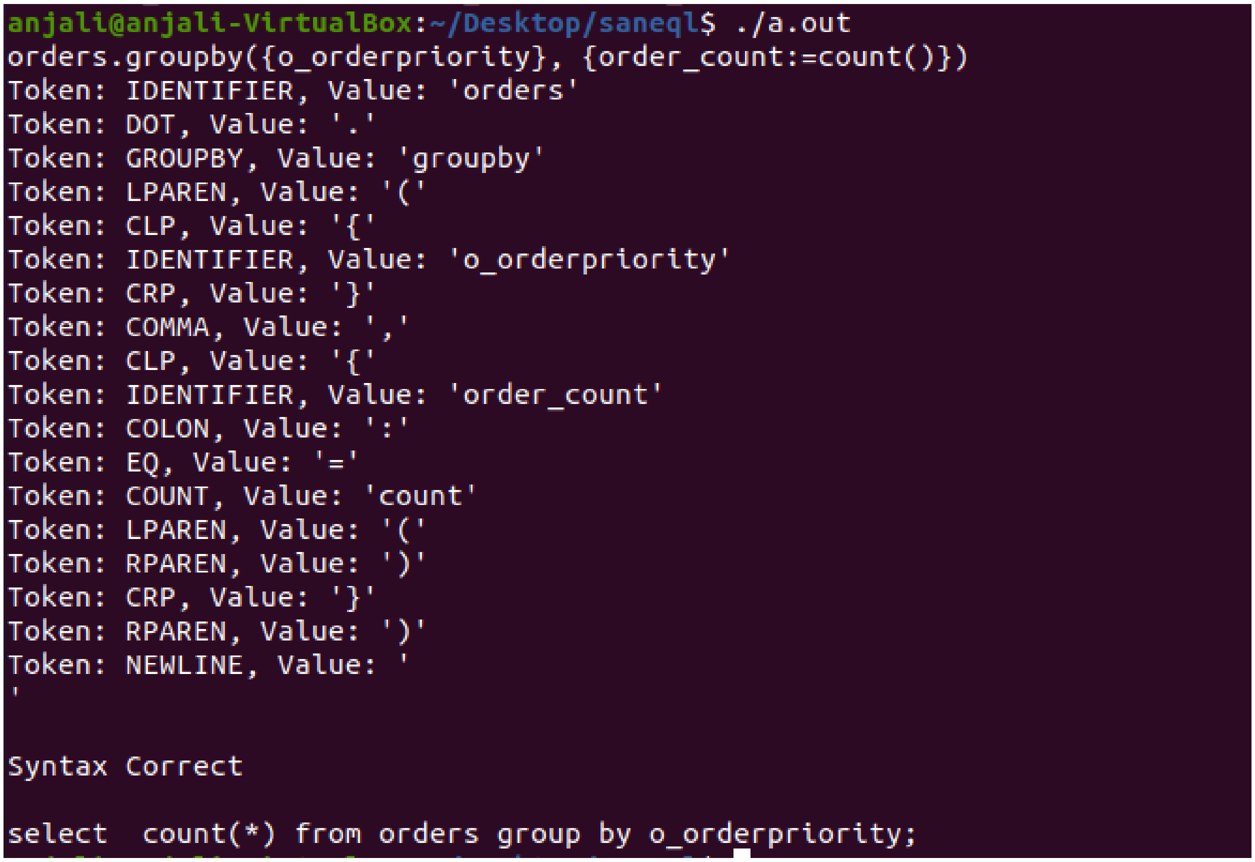


```
anjali@anjali-VirtualBox:~/Desktop/saneql$ ./a.out
orders.join(customer, c_custkey=o_custkey)
Token: IDENTIFIER, Value: 'orders'
Token: DOT, Value: '.'
Token: JOIN, Value: 'join'
Token: LPAREN, Value: '('
Token: IDENTIFIER, Value: 'customer'
Token: COMMA, Value: ','
Token: IDENTIFIER, Value: 'c_custkey'
Token: EQ, Value: '='
Token: IDENTIFIER, Value: 'o_custkey'
Token: RPAREN, Value: ')'
Token: NEWLINE, Value: '
'

Syntax Correct

select * from orders inner join customer on c_custkey=o_custkey;
anjali@anjali-VirtualBox:~/Desktop/saneql$
```

SaneQL Input: orders.join(customer, c_custkey=o_custkey)

SQL Output: select * from orders inner join customer on c_custkey=o_custkey;

Groupby Query

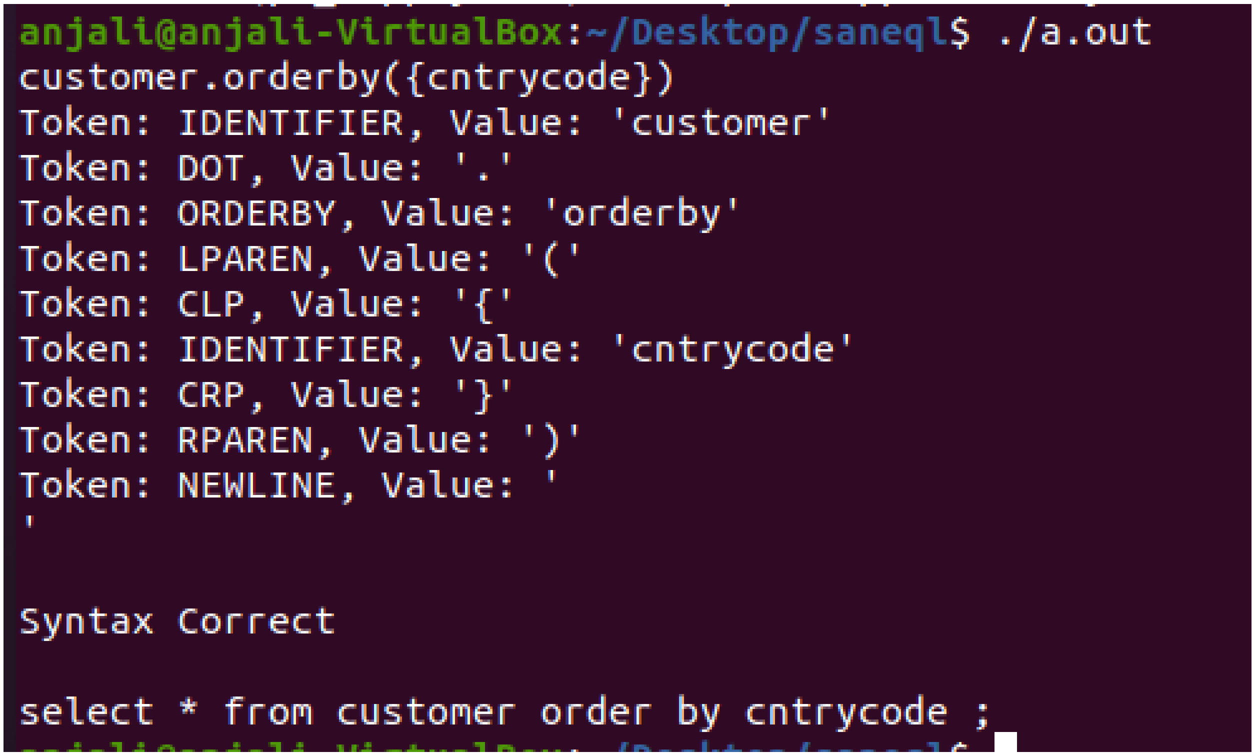


```
anjali@anjali-VirtualBox:~/Desktop/saneql$ ./a.out
orders.groupby({o_orderpriority}, {order_count:=count()})
Token: IDENTIFIER, Value: 'orders'
Token: DOT, Value: '.'
Token: GROUPBY, Value: 'groupby'
Token: LPAREN, Value: '('
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'o_orderpriority'
Token: CRP, Value: '}'
Token: COMMA, Value: ','
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'order_count'
Token: COLON, Value: ':'
Token: EQ, Value: '='
Token: COUNT, Value: 'count'
Token: LPAREN, Value: '('
Token: RPAREN, Value: ')'
Token: CRP, Value: '}'
Token: RPAREN, Value: ')'
Token: NEWLINE, Value: '
'


Syntax Correct

select  count(*) from orders group by o_orderpriority;
```

SaneQL Input: orders.groupby({o_orderpriority}, {order_count:=count()})

SQL Output: select count(*) from orders group by o_orderpriority;

Orderby Query

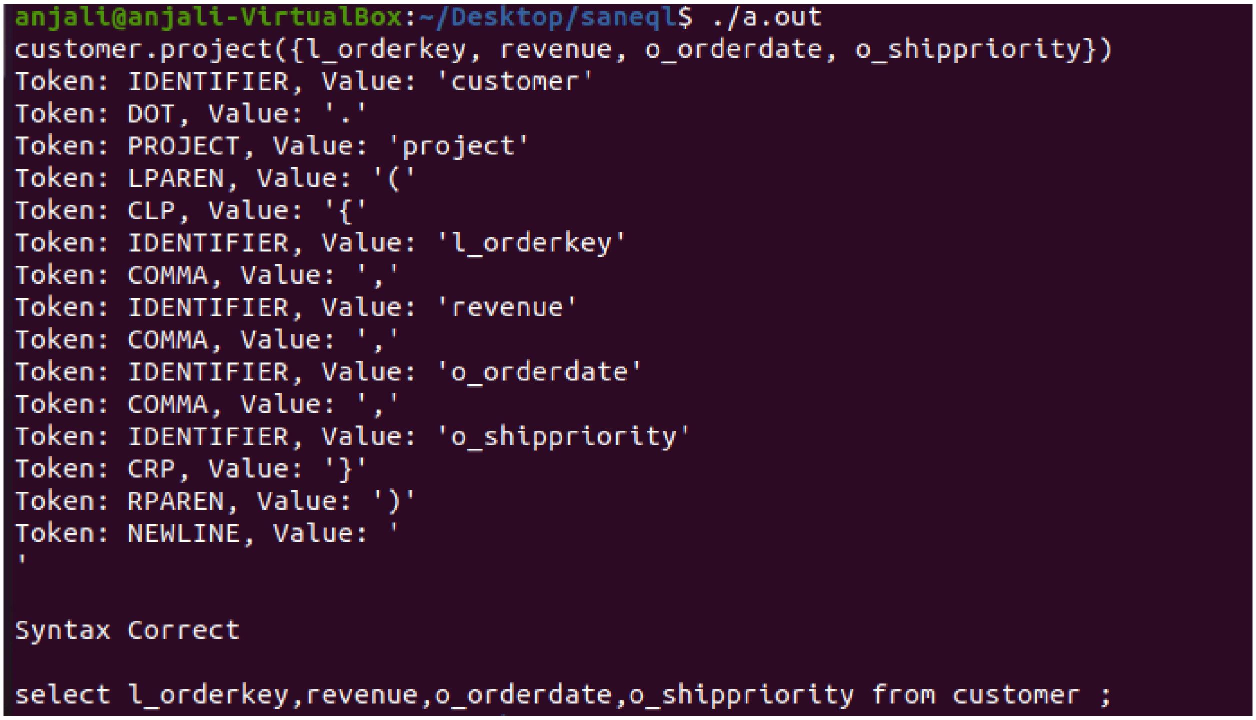


```
anjali@anjali-VirtualBox:~/Desktop/saneql$ ./a.out
customer.orderby({cntrycode})
Token: IDENTIFIER, Value: 'customer'
Token: DOT, Value: '.'
Token: ORDERBY, Value: 'orderby'
Token: LPAREN, Value: '('
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'cntrycode'
Token: CRP, Value: '}'
Token: RPAREN, Value: ')'
Token: NEWLINE, Value: '
'


Syntax Correct

select * from customer order by cntrycode ;
```

SaneQL Input: customer.orderby({cntrycode})

SQL Output: select * from customer order by cntrycode ;

Project Query

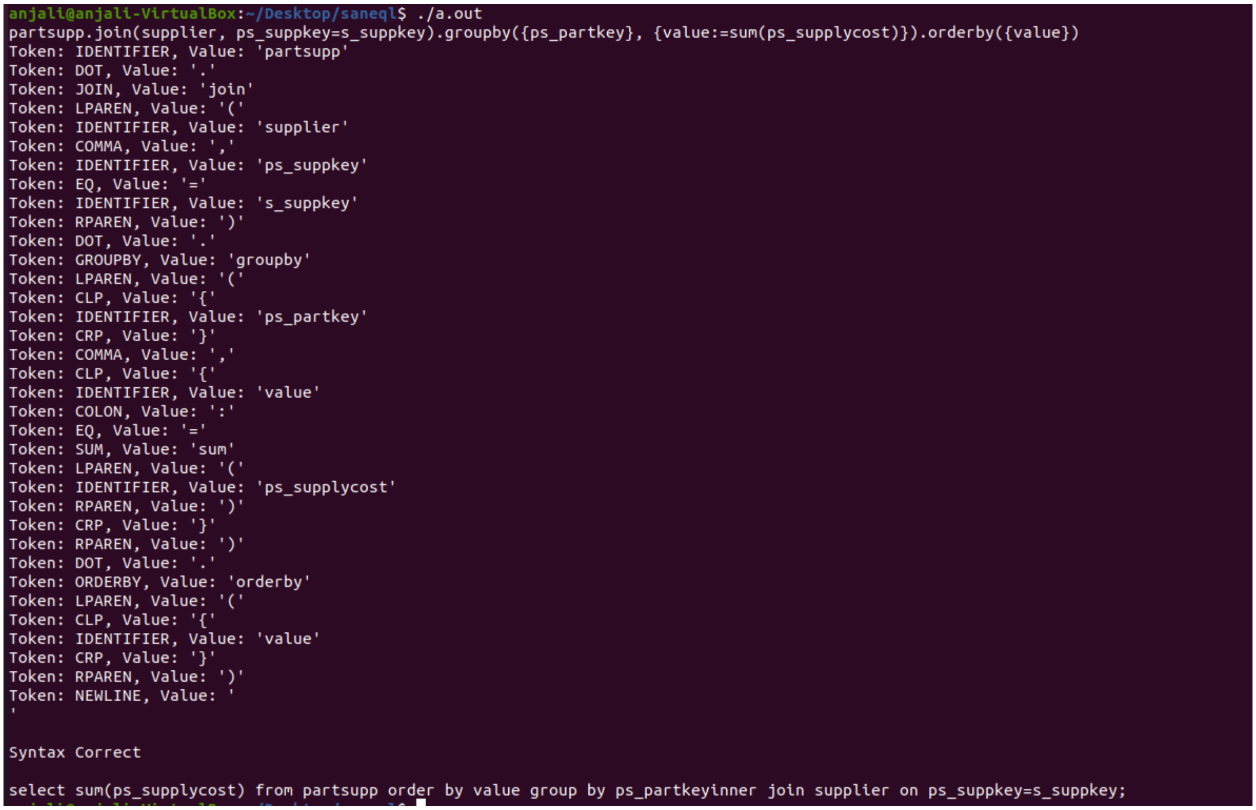


```
anjali@anjali-VirtualBox:~/Desktop/saneql$ ./a.out
customer.project({l_orderkey, revenue, o_orderdate, o_shippriority})
Token: IDENTIFIER, Value: 'customer'
Token: DOT, Value: '.'
Token: PROJECT, Value: 'project'
Token: LPAREN, Value: '('
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'l_orderkey'
Token: COMMA, Value: ','
Token: IDENTIFIER, Value: 'revenue'
Token: COMMA, Value: ','
Token: IDENTIFIER, Value: 'o_orderdate'
Token: COMMA, Value: ','
Token: IDENTIFIER, Value: 'o_shippriority'
Token: CRP, Value: '}'
Token: RPAREN, Value: ')'
Token: NEWLINE, Value: '
'

Syntax Correct

select l_orderkey,revenue,o_orderdate,o_shippriority from customer ;
```

SaneQL Input: customer.project({l_orderkey, revenue, o_orderdate, o_shippriority})

SQL Output: select l_orderkey,revenue,o_orderdate,o_shippriority from customer ;

Groupby-Join Query

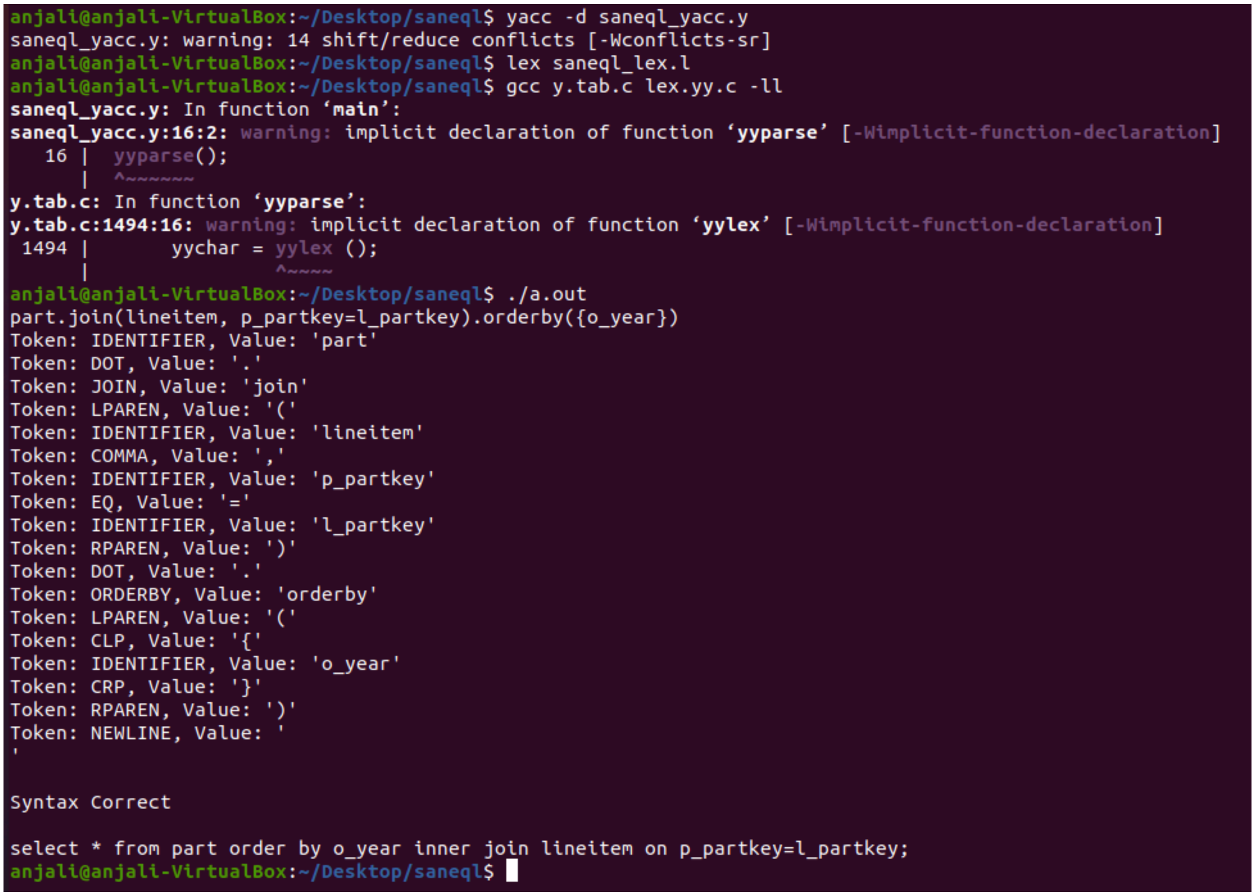


```
anjali@anjali-VirtualBox:~/Desktop/saneql$ ./a.out
partsupp.join(supplier, ps_suppkey=s_suppkey).groupby({ps_partkey}, {value:=sum(ps_supplycost)}).orderby({value})
Token: IDENTIFIER, Value: 'partsupp'
Token: DOT, Value: '.'
Token: JOIN, Value: 'join'
Token: LPAREN, Value: '('
Token: IDENTIFIER, Value: 'supplier'
Token: COMMA, Value: ','
Token: IDENTIFIER, Value: 'ps_suppkey'
Token: EQ, Value: '='
Token: IDENTIFIER, Value: 's_suppkey'
Token: RPAREN, Value: ')'
Token: DOT, Value: '.'
Token: GROUPBY, Value: 'groupby'
Token: LPAREN, Value: '('
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'ps_partkey'
Token: CRP, Value: '}'
Token: COMMA, Value: ','
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'value'
Token: COLON, Value: ':'
Token: EQ, Value: '='
Token: SUM, Value: 'sum'
Token: LPAREN, Value: '('
Token: IDENTIFIER, Value: 'ps_supplycost'
Token: RPAREN, Value: ')'
Token: CRP, Value: '}'
Token: RPAREN, Value: ')'
Token: DOT, Value: '.'
Token: ORDERBY, Value: 'orderby'
Token: LPAREN, Value: '('
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'value'
Token: CRP, Value: '}'
Token: RPAREN, Value: ')'
Token: NEWLINE, Value: '
'

Syntax Correct

select sum(ps_supplycost) from partsupp order by value group by ps_partkeyinner join supplier on ps_suppkey=s_suppkey;
```

SaneQL Input: part.join(lineitem, p_partkey=l_partkey).orderby({o_year})

SQL Output: select * from part order by o_year inner join lineitem on p_partkey=l_partkey;

Join-Orderby-Groupby Query



```
anjali@anjali-VirtualBox:~/Desktop/saneql$ yacc -d saneql_yacc.y
saneql_yacc.y: warning: 14 shift/reduce conflicts [-Wconflicts-sr]
anjali@anjali-VirtualBox:~/Desktop/saneql$ lex saneql_lex.l
anjali@anjali-VirtualBox:~/Desktop/saneql$ gcc y.tab.c lex.yy.c -ll
saneql_yacc.y: In function ‘main’:
saneql_yacc.y:16:2: warning: implicit declaration of function ‘yyparse’ [-Wimplicit-function-declaration]
   16 |  yyparse();
      |  ^~~~~~~
y.tab.c: In function ‘yyparse’:
y.tab.c:1494:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]
 1494 |       yychar = yylex ();
      |                ^~~~~
anjali@anjali-VirtualBox:~/Desktop/saneql$ ./a.out
part.join(lineitem, p_partkey=l_partkey).orderby({o_year})
Token: IDENTIFIER, Value: 'part'
Token: DOT, Value: '.'
Token: JOIN, Value: 'join'
Token: LPAREN, Value: '('
Token: IDENTIFIER, Value: 'lineitem'
Token: COMMA, Value: ','
Token: IDENTIFIER, Value: 'p_partkey'
Token: EQ, Value: '='
Token: IDENTIFIER, Value: 'l_partkey'
Token: RPAREN, Value: ')'
Token: DOT, Value: '.'
Token: ORDERBY, Value: 'orderby'
Token: LPAREN, Value: '('
Token: CLP, Value: '{'
Token: IDENTIFIER, Value: 'o_year'
Token: CRP, Value: '}'
Token: RPAREN, Value: ')'
Token: NEWLINE, Value: '
'

Syntax Correct

select * from part order by o_year inner join lineitem on p_partkey=l_partkey;
anjali@anjali-VirtualBox:~/Desktop/saneql$ 
```

SaneQL Input: partsupp.join(supplier, ps_suppkey=s_suppkey).groupby({ps_partkey}, {value:=sum(ps_supplycost)}).orderby({value})

SQL Output: select sum(ps_supplycost) from partsupp order by value group by ps_partkeyinner join supplier on ps_suppkey=s_suppkey;