REPORT

Name: Avaneesh Lingwal
Roll Number : 12
Topic: DELETE PARSER

Syntax:

According to MySQL standard, the syntax of DELETE is:

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name [[AS] tbl_alias]
    [PARTITION (partition_name [, partition_name] ...)]
    [WHERE where_condition]
    [ORDER BY ...]
    [LIMIT row_count]
```

Patterns:

Name definitions:

```
digit [0-9]
alphabets [a-zA-Z]
alphanum ({alphabets}|{digit}|"_"|"$")
identifier ({digit}[a-zA-Z_$]|[a-zA-Z_$])({alphanum}*)
```

The identifier according to MySQL standard can start from the alphabet or digit, but cannot consist only of digits.

Rules:

Patterns	Actions	Rationale
`` \`[^\`]+\` ``	`return IDENTIFIER;`	Anything inside the backticks is used to define quoted identifiers by MySQL.
`(d|D)(e|E)(l|L)(e|E)(t|T)(e|E)`	`return DELETE;`	Case Insensitive DELETE..
`(f|F)(r|R)(o|O)(m|M)`	`return FROM;`	Case Insensitive FROM.
`(w|W)(h|H)(e|E)(r|R)(e|E)`	`return WHERE;`	Case Insensitive WHERE.
`(a|A)(s|S)`	`return AS;`	Case Insensitive AS.
`(a|A)(n|N)(d|D)|(o|O)(r|R)`	`return CONDITIONAL_OP;`	Case Insensitive AND | OR.
		CASE Insensitive NOT. Not included in

`(n|N)(o|O)(t|T)`	`return NOT;`	CONDITIONAL_OP, as it is a unary operator. Ex: DELETE FROM TABLE WHERE NOT name = “something”;
`(l|L)(i|I)(k|K)(e|E)`	`return RELATIONAL_OP;`	Case Insensitive LIKE.
`{digit}+`	`return NUMBER;`	For numbers.
`[\"](.)+[\"]`	`return TEXT;`	For text.
`[\'](.)+[\']`	`return TEXT;`	For text.
`"<"`	`return RELATIONAL_OP;`	Less than.
`">"`	`return RELATIONAL_OP;`	Greater than.
`"<="`	`return RELATIONAL_OP;`	Less than or equal.
`">="`	`return RELATIONAL_OP;`	Greater than or equal.

`"="`	`return RELATIONAL_OP;`	Equality.
`"!="`	`return RELATIONAL_OP;`	Not equal.
`“;”`	`return SEMICOLON;`	Semi-colon.
`{identifier}`	`return IDENTIFIER;`	Identifier.
`{identifier}["."]{identifier}`	`return IDENTIFIER;`	For the case where table_name.column_name
`\n`	`return NEWLINE;`	For Newline.
`[ \t]|" "`	`;`	Escaping tab and spaces.
`.`	`return *yytext;`	Returning text.

Tokens:

```
%token DELETE FROM IDENTIFIER WHERE CONDITIONAL_OP RELATIONAL_OP SEMICOLON
TEXT NUMBER NEWLINE AS NOT
```

Grammar:

```
%%
```

```
/*
   Initial Rule.
*/
line: delete {printf("Syntax Correct\n");

   return 0;

};



delete : DELETE from | error {yyerror(" : Did you mean \"DELETE\" ? \n");
return 1; };

/*
   Covering two cases of deletion from a single table.
   1. When a condition is specified using a WHERE clause.
   2. When no condition is specified.
*/

from : FROM table where | FROM table semicolon NEWLINE |  error {yyerror("
: Did you mean \" FROM \" ? \n"); return 1; };

/*
   Covering two cases of table name:
   1. Without any alias.
   2. With an alias using AS keyword.

*/

table : IDENTIFIER | IDENTIFIER AS IDENTIFIER | error {yyerror(" : table
name is missing.\n"); return 1; };



where : WHERE condition semicolon NEWLINE |
error {yyerror(" : Did you mean \" WHERE \" ? \n"); return 1; };
```

```
/*
   Covering the cases of different conditions that can be specified.

   1 - 4 are self explanatory.
   5 - 8 covers the cases where rules [1,4] are appended with a
conditional operator and another condition.
   9 covers the case where we can apply the NOT operator in front of a
conditional statement.

*/

condition : IDENTIFIER RELATIONAL_OP IDENTIFIER |
           IDENTIFIER RELATIONAL_OP TEXT  |
           IDENTIFIER RELATIONAL_OP NUMBER |
           NUMBER RELATIONAL_OP NUMBER |
           IDENTIFIER RELATIONAL_OP IDENTIFIER CONDITIONAL_OP condition |
           IDENTIFIER RELATIONAL_OP TEXT CONDITIONAL_OP condition |
           IDENTIFIER RELATIONAL_OP NUMBER CONDITIONAL_OP condition |
           NUMBER RELATIONAL_OP NUMBER CONDITIONAL_OP condition |
           NOT condition |

           error {

               yyerror(" : Incorrect Condition \n");
               return 1;
           };

semicolon : SEMICOLON | error {yyerror(" : Missing semicolon \";\" \n");
return 1; };
```

Cases Covered:

Basic syntax: Handles the fundamental case of:
    DELETE FROM TABLE WHERE CONDITION structure.
Quoted and Unquoted Identifiers: Handles both quoted and unquoted identifiers.

Case Insensitivity is also handled.

Table Aliases using AS clause are also handled.
Basic Conditions and **Compound Conditions** joined by AND, OR and NOT.

Cases Not Covered:

Multi-Table DELETES.
LIMIT, ORDER BY and JOIN clauses.
IN, BETWEEN, IS NULL , IS NOT NULL is not covered.
Parentheses, Arithmetic, Functions and Subqueries.
Modifiers like LOW_PRIORITY, QUICK and IGNORE.

Code:

delete.l

```
/* Defining neccessary definitions. */
/*  So only digit cannot be an identifier, must have any other character
after it, but identifier can be of only alphabet. */


digit [0-9]
alphabets [a-zA-Z]
alphanum ({alphabets}|{digit}|"_"|"$")
identifier ({digit}[a-zA-Z_$]|[a-zA-Z_$])({alphanum}*)

%{
   #include <stdio.h>
   #include "y.tab.h"

%}



%%

\`[^\`]+\`      return IDENTIFIER; // If it is quoted, then anything inside
it can be identifier.
```

```
(d|D)(e|E)(l|L)(e|E)(t|T)(e|E)  return DELETE;

(f|F)(r|R)(o|O)(m|M)   return FROM;

(w|W)(h|H)(e|E)(r|R)(e|E) return WHERE;

(a|A)(s|S)    return AS;

(a|A)(n|N)(d|D)|(o|O)(r|R) return CONDITIONAL_OP;  /* NOT is not included,
because it is a unary operator and to handle the case where the

                               conditions can be specified as (NOT
condition).  */


(n|N)(o|O)(t|T) return NOT;

(l|L)(i|I)(k|K)(e|E)    return RELATIONAL_OP;

{digit}+     return NUMBER;

[\"](.)+[\"] return TEXT;

[\'](.)+[\'] return TEXT;

"<"         return RELATIONAL_OP;
">"          return RELATIONAL_OP;
"<="         return RELATIONAL_OP;
">="         return RELATIONAL_OP;
"="         return RELATIONAL_OP;
"!="         return RELATIONAL_OP;

";"          return SEMICOLON;

{identifier} return IDENTIFIER;



{identifier}["."]{identifier}  return IDENTIFIER; // To handle the case
where table_name.column_name
```

```
\n          return NEWLINE;


[ \t]|" "   ;

.           return *yytext;

%%
```

delete.y

```
/*

Including the neccessary headers.

*/
%{

   #include <stdio.h>
   #include <stdlib.h>
   void yyerror(const char* s);
   int yylex(void);


%}

/*

   Neccessary tokens.

*/

%token DELETE FROM IDENTIFIER WHERE CONDITIONAL_OP RELATIONAL_OP SEMICOLON
TEXT NUMBER NEWLINE AS NOT
```

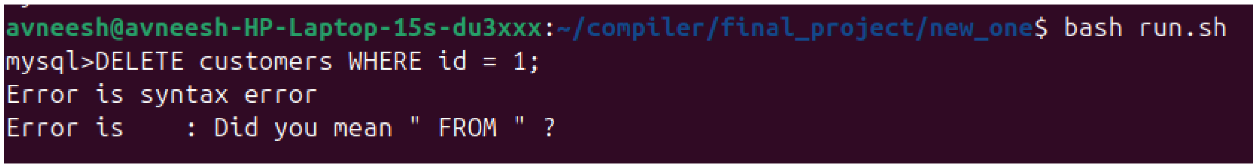
```
%%
/*  Grammar Rules:

   line : delete

   delete : DELETE from | error

   from :  FROM table where | FROM table semicolon NEWLINE | error

   table : IDENTIFIER | IDENTIFIER AS IDENTIFIER | error

   where: WHERE condition semicolon NEWLINE | error

   condition : IDENTIFIER RELATIONAL_OP IDENTIFIER

       | IDENTIFIER RELATIONAL_OP TEXT

       | IDENTIFIER RELATIONAL_OP NUMBER

       | IDENTIFIER RELATIONAL_OP IDENTIFIER CONDITIONAL_OP condition

       | IDENTIFIER RELATIONAL_OP TEXT CONDITIONAL_OP condition

       | IDENTIFIER RELATIONAL_OP NUMBER CONDITIONAL_OP condition

       | NUMBER RELATIONAL_OP NUMBER

       | NUMBER RELATIONAL_OP NUMBER CONDITIONAL_OP condition

       | NOT condition

       | error


*/
```

```
/*
   Initial Rule.
*/
line: delete {printf("Syntax Correct\n");

   return 0;

};



delete : DELETE from | error {yyerror(" : Did you mean \"DELETE\" ? \n");
return 1; };

/*
   Covering two cases of deletion from a single table.
   1. When a condition is specified using a WHERE clause.
   2. When no condition is specified.
*/

from : FROM table where | FROM table semicolon NEWLINE |  error {yyerror("
: Did you mean \" FROM \" ? \n"); return 1; };

/*
   Covering two cases of table name:
   1. Without any alias.
   2. With an alias using AS keyword.

*/

table : IDENTIFIER | IDENTIFIER AS IDENTIFIER | error {yyerror(" : table
name is missing.\n"); return 1; };



where : WHERE condition semicolon NEWLINE |
error {yyerror(" : Did you mean \" WHERE \" ? \n"); return 1; };

/*
```

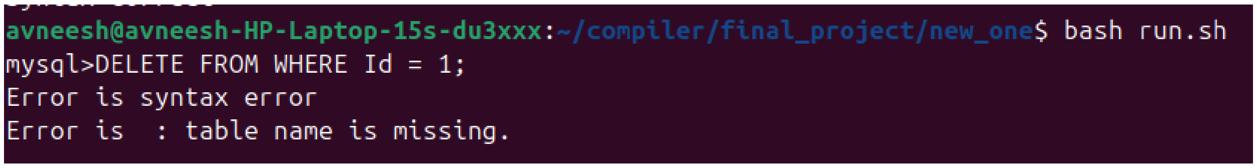
```
   Covering the cases of different conditions that can be specified.

   1 - 4 are self explainatory.
   5 - 8 covers the cases where rules [1,4] are appended with an
conditional operator and another condition.
   9 covers the case where we can apply NOT operator in front of a
conditional statement.

*/

condition : IDENTIFIER RELATIONAL_OP IDENTIFIER |
           IDENTIFIER RELATIONAL_OP TEXT |
           IDENTIFIER RELATIONAL_OP NUMBER |
           NUMBER RELATIONAL_OP NUMBER |
           IDENTIFIER RELATIONAL_OP IDENTIFIER CONDITIONAL_OP condition |
           IDENTIFIER RELATIONAL_OP TEXT CONDITIONAL_OP condition |
           IDENTIFIER RELATIONAL_OP NUMBER CONDITIONAL_OP condition |
           NUMBER RELATIONAL_OP NUMBER CONDITIONAL_OP condition |
           NOT condition |

           error {

               yyerror(" : Incorrect Condtion \n");
               return 1;
           };

semicolon : SEMICOLON | error {yyerror(" : Missing semicolon \";\" \n");
return 1; };


%%

int main(void){

   printf("mysql>");
   yyparse();
   return 0;

}
```

```
void yyerror(const char* s){

   printf("Error is %s\n", s);

}

int yywrap(){
   return 1;
}
```

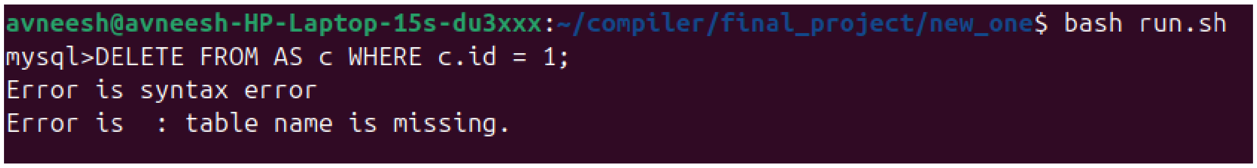
Test Cases:



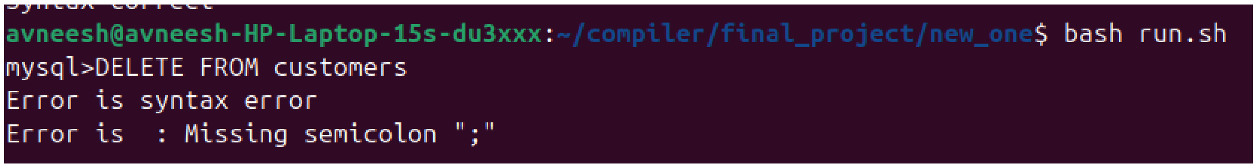
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE customers WHERE id = 1;
Error is syntax error
Error is    : Did you mean " FROM " ?
```



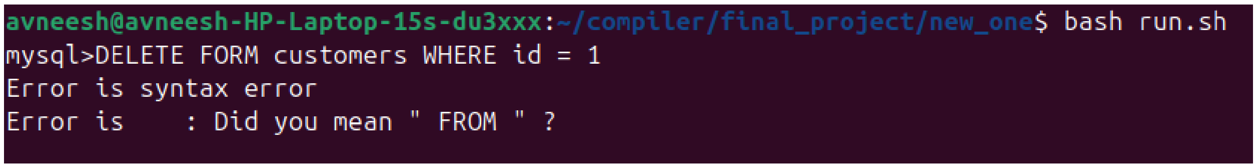
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM WHERE Id = 1;
Error is syntax error
Error is  : table name is missing.
```



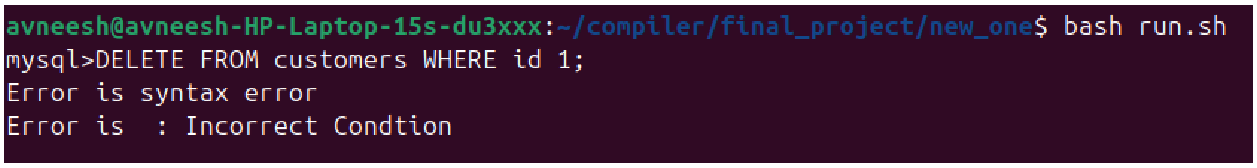
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM AS c WHERE c.id = 1;
Error is syntax error
Error is  : table name is missing.
```



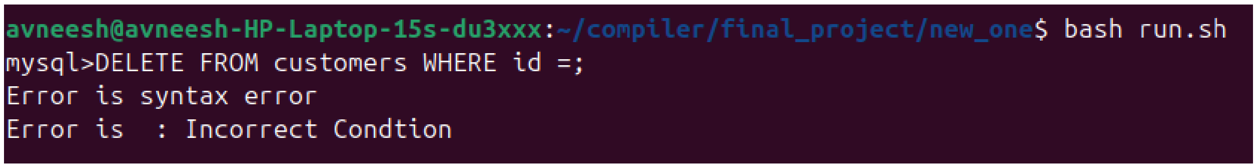
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers
Error is syntax error
Error is  : Missing semicolon ";"
```



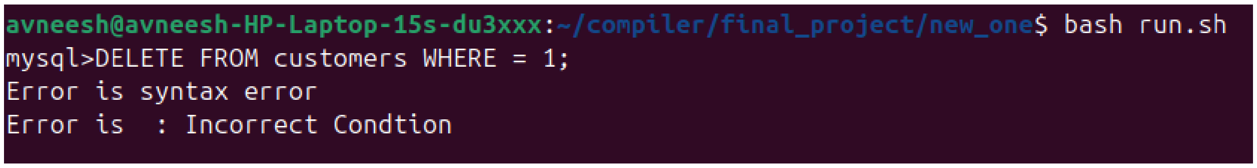
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FORM customers WHERE id = 1
Error is syntax error
Error is    : Did you mean " FROM " ?
```



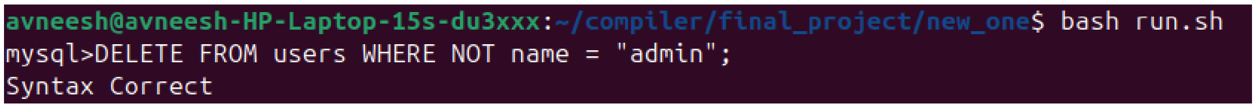
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers WHERE id 1;
Error is syntax error
Error is  : Incorrect Condtion
```



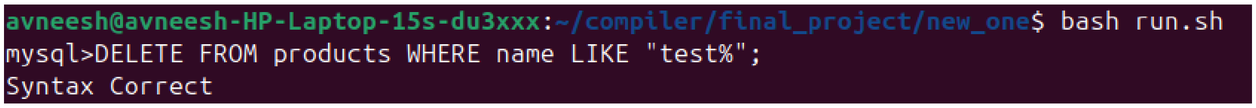
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers WHERE id =;
Error is syntax error
Error is  : Incorrect Condtion
```



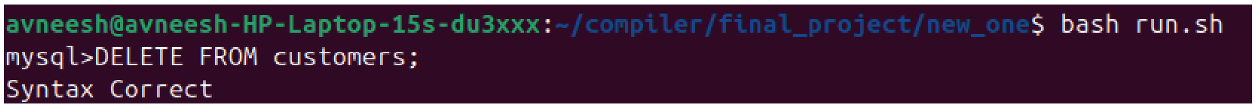
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers WHERE = 1;
Error is syntax error
Error is  : Incorrect Condtion
```



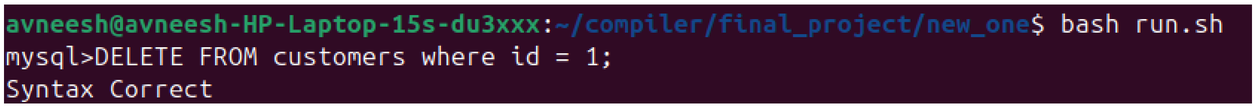
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM users WHERE NOT name = "admin";
Syntax Correct
```



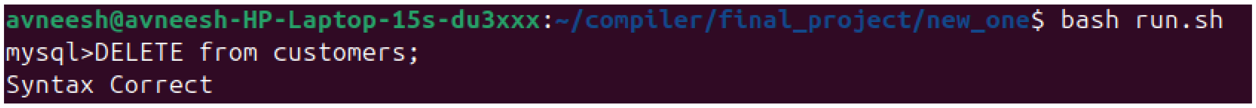
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM products WHERE name LIKE "test%";
Syntax Correct
```



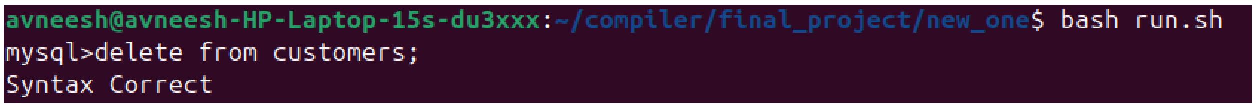
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers;
Syntax Correct
```



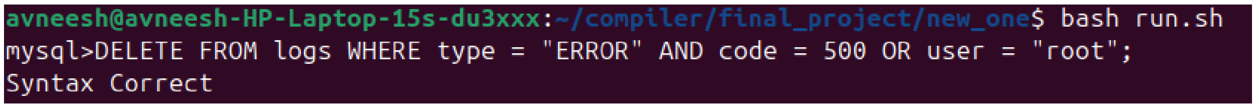
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers where id = 1;
Syntax Correct
```



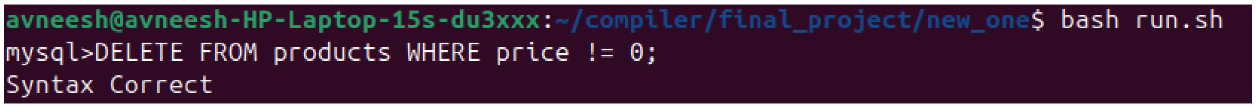
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE from customers;
Syntax Correct
```



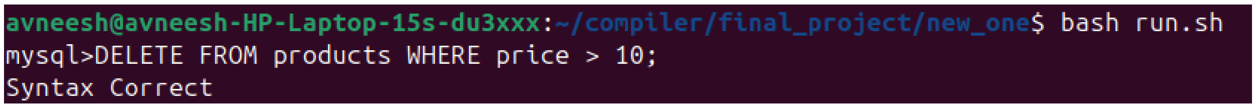
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>delete from customers;
Syntax Correct
```



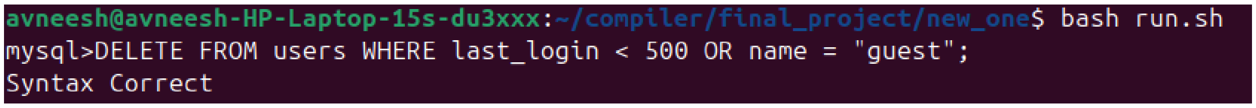
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM logs WHERE type = "ERROR" AND code = 500 OR user = "root";
Syntax Correct
```



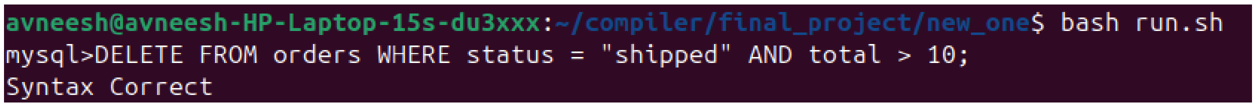
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM products WHERE price != 0;
Syntax Correct
```



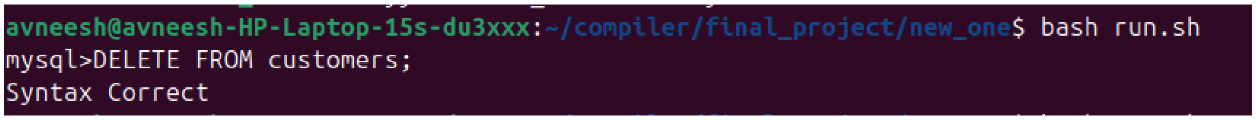
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM products WHERE price > 10;
Syntax Correct
```



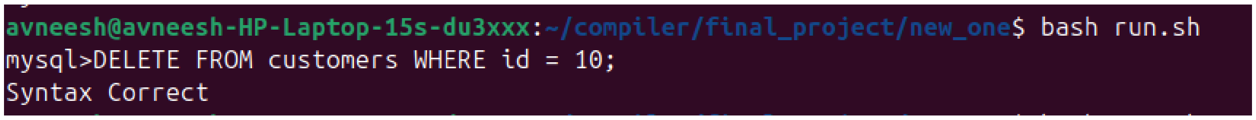
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM users WHERE last_login < 500 OR name = "guest";
Syntax Correct
```



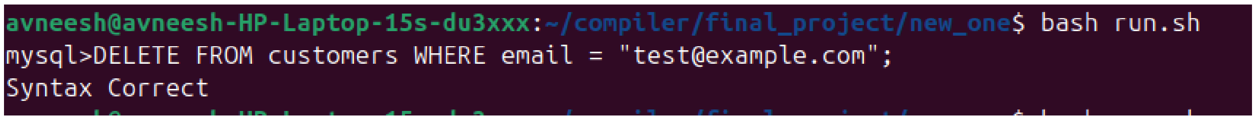
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM orders WHERE status = "shipped" AND total > 10;
Syntax Correct
```



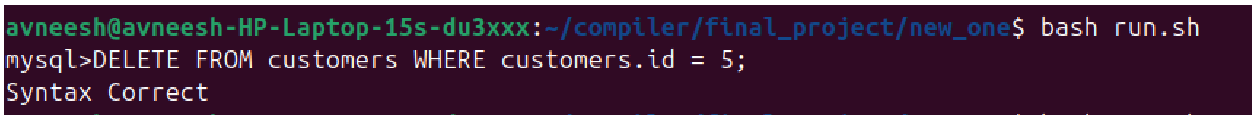
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers;
Syntax Correct
```



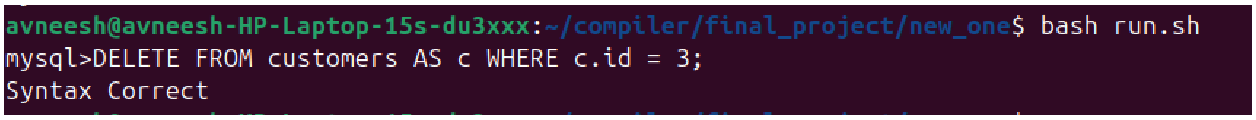
```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers WHERE id = 10;
Syntax Correct
```



```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers WHERE email = "test@example.com";
Syntax Correct
```



```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers WHERE customers.id = 5;
Syntax Correct
```



```
avneesh@avneesh-HP-Laptop-15s-du3xxx:~/compiler/final_project/new_one$ bash run.sh
mysql>DELETE FROM customers AS c WHERE c.id = 3;
Syntax Correct
```