Assignment 2

course_code: COMP4003

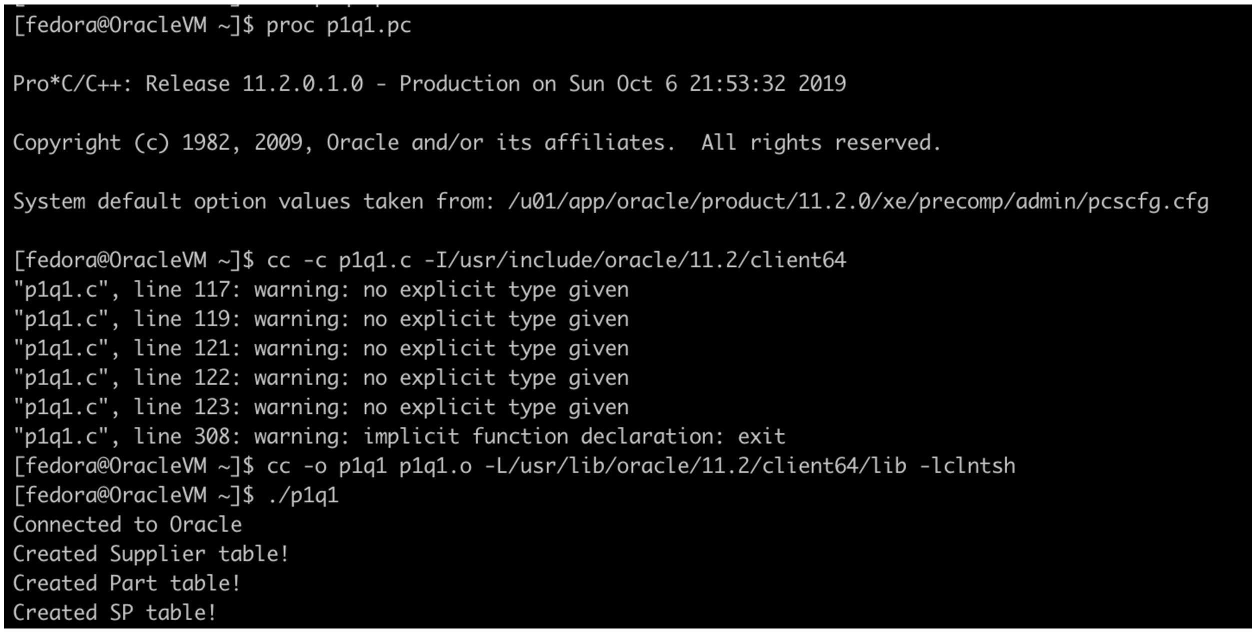
name: Emma Orhun

student_num: 101071651

date_completed: Oct. 5th 2019

1. P1Q1.pc

```
#include <stdio.h>
#include <string.h>

exec sql include sqlca;

exec sql begin declare section;
  char *identity = "fedora/oracle";
  char sqlstmt[1024];
exec sql end declare section;

int main(){
  exec sql connect :identity;
  if (sqlca.sqlcode < 0){
    printf("Could not connect to oracle!\n");
    exit(1);
  }
  else{
    printf("Connected to Oracle\n");
  }

  exec sql set transaction read write;

  //Drop tables if they exist..
  exec sql execute immediate "DROP TABLE Supplier";
  exec sql execute immediate "DROP TABLE Part";
  exec sql execute immediate "DROP TABLE SP";


  //Creat Tables
  //Create the Supplier table
  strcpy(sqlstmt,"CREATE TABLE Supplier (S# char(2) primary key, sname varchar(10), status integer, city varchar(10))");

  exec sql execute immediate :sqlstmt;
  if (sqlca.sqlcode < 0){
    printf("Could not create Supplier table\n");
    exit(1);
  }
  else{
    printf("Created Supplier table!\n");
  }

  //Create the Part table
  strcpy(sqlstmt,"CREATE TABLE Part (P# char(2) primary key, pname varchar(10), color varchar(10), weight decimal(3,1), city varchar

  exec sql execute immediate :sqlstmt;
  if (sqlca.sqlcode < 0){
    printf("Could not create Part table\n");
    exit(1);
  }
  else{
    printf("Created Part table!\n");
  }

  //Create the SP lookup table
  strcpy(sqlstmt,"CREATE TABLE SP (S# char(2) not null, P# char(2) not null, foreign key (S#) references Supplier(S#), foreign key (

  exec sql execute immediate :sqlstmt;
  if (sqlca.sqlcode < 0){
    printf("Could not create SP lookup table\n");
    exit(1);
  }
  else{
    printf("Created SuppToPart table!\n");
  }
exec sql commit release;
exit(0);
}
```

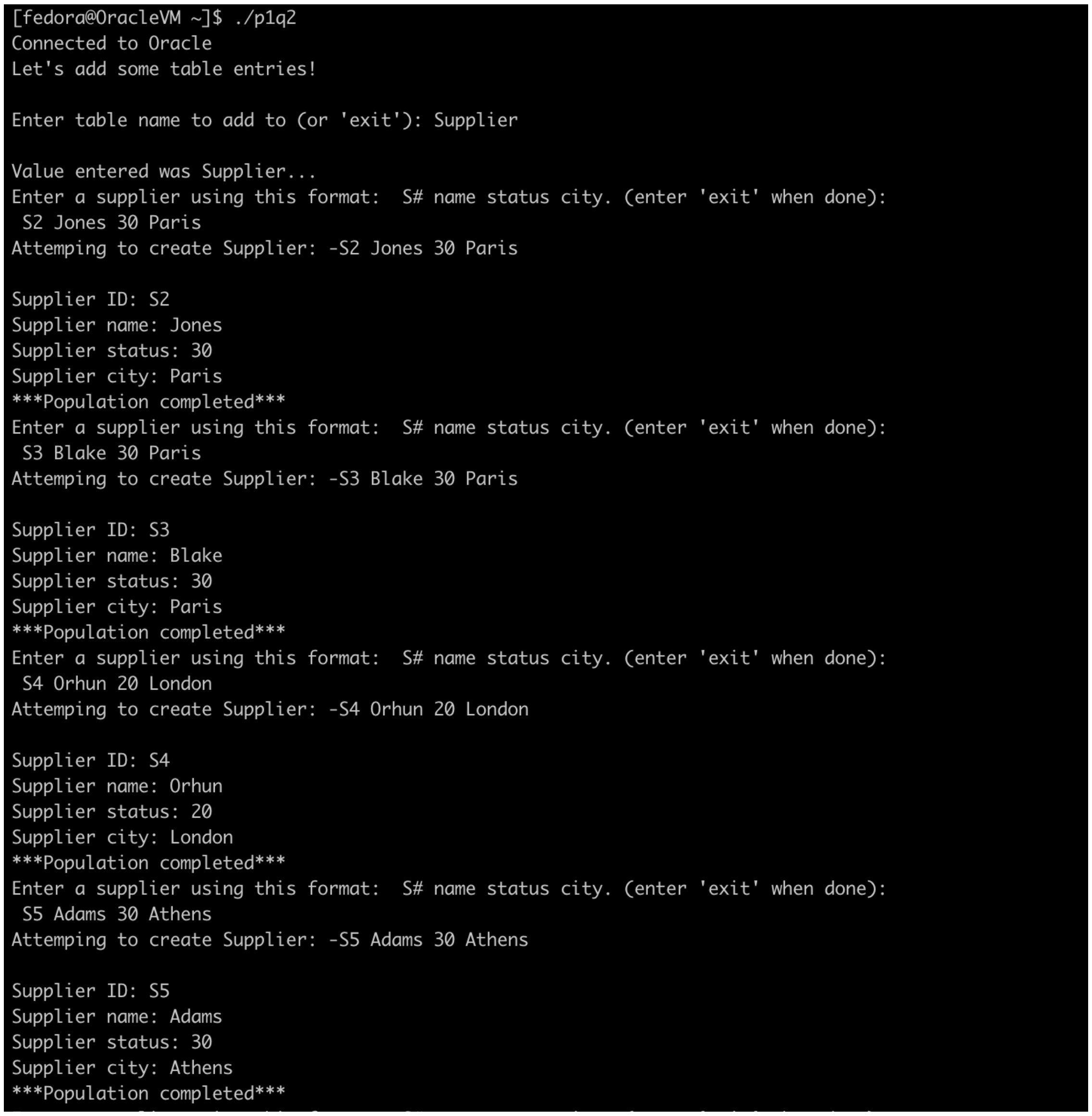


```
[fedora@OracleVM ~]$ proc p1q1.pc

Pro*C/C++: Release 11.2.0.1.0 - Production on Sun Oct 6 21:53:32 2019

Copyright (c) 1982, 2009, Oracle and/or its affiliates.  All rights reserved.

System default option values taken from: /u01/app/oracle/product/11.2.0/xe/precomp/admin/pcscfg.cfg

[fedora@OracleVM ~]$ cc -c p1q1.c -I/usr/include/oracle/11.2/client64
"p1q1.c", line 117: warning: no explicit type given
"p1q1.c", line 119: warning: no explicit type given
"p1q1.c", line 121: warning: no explicit type given
"p1q1.c", line 122: warning: no explicit type given
"p1q1.c", line 123: warning: no explicit type given
"p1q1.c", line 308: warning: implicit function declaration: exit
[fedora@OracleVM ~]$ cc -o p1q1 p1q1.o -L/usr/lib/oracle/11.2/client64/lib -lclntsh
[fedora@OracleVM ~]$ ./p1q1
Connected to Oracle
Created Supplier table!
Created Part table!
Created SP table!
```

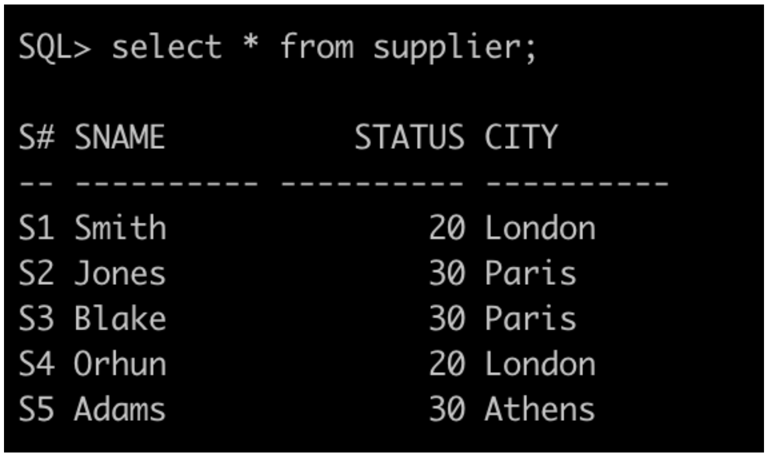
2.

```
#include <stdio.h>
#include <string.h>

exec sql include sqlca;

exec sql begin declare section;
  char *identity = "fedora/oracle";
  char statement[1024];
exec sql end declare section;

void populatePartTable(){
  char *input;
  char partInput[1024];

  while(1){
    printf("Enter a part using this format: P# name colour weight city.  (enter exit when done):\n");
    fgets(partInput, 1024, stdin);
    printf("Attemping to create Part: -%s\n", partInput);

    if( strcmp(partInput, "exit") == 0){
      printf("Exitting\n");
      break;
    }
    char *partID = strtok(partInput, " \n");
printf("Extracted ID: %s\n", partID);
    char *partName = strtok(NULL, " \n");
printf("Extracted Name: %s\n", partName);
    char *partColour = strtok(NULL, " \n");
printf("Extracted Color: %s\n", partColour);
    float weight = atof(strtok(NULL, " \n"));
printf("Extracted weight: %f\n", weight);
```

```
    char *city = strtok (NULL, " \n");
printf("Extracted city: %s\n", city);

    strcpy(statement, "INSERT INTO part VALUES (:v1, :v2, :v3, :v4, :v5)");
    exec sql prepare s from :statement;
    if (sqlca.sqlcode < 0){
      printf("Statement not prepared!\n");
      break;
    }

    exec sql execute s using :partID, :partName, :partColour, :weight, :city;
    if (sqlca.sqlcode < 0){
      printf("Population not complete!\n");
    }
    else {
      printf("***Population completed***\n");
    }
  }
}

void populateSupplierTable(){
  char *input;
  char supplierInput[1024];

  while(1){
    printf("Enter a supplier using this format:  S# name status city. (enter 'exit' when done):\n ");
    fgets(supplierInput, 1024, stdin);
    printf("Attemping to create Supplier: -%s\n", supplierInput);
    supplierInput[strlen(supplierInput) - 1] = 0;

    if( strcmp(supplierInput, "exit") == 0){
      printf("Exitting\n");
      break;
    }
    char *supplierID = strtok(supplierInput, " \n");
printf("Supplier ID: %s\n", supplierID);
    char *supplierName = strtok(NULL, " \n");
printf("Supplier name: %s\n", supplierName);
    int status = atoi(strtok(NULL, " \n"));
printf("Supplier status: %d\n", status);
    char *city = strtok(NULL, " \n");
printf("Supplier city: %s\n", city);

    strcpy(statement, "INSERT INTO supplier VALUES (:v1, :v2, :v3, :v4)");
    exec sql prepare s from :statement;
    if (sqlca.sqlcode < 0){
      printf("Statement not prepared!\n");
      break;
    }

    exec sql execute s using :supplierID, :supplierName, :status, :city;
    if (sqlca.sqlcode < 0){
      printf("Population not complete!\n");
    }
    else {
      printf("***Population completed***\n");
    }
  }
}

void populateSPTable(){
  char *input;
  char entryInput[1024];

  while(1){
    printf("Enter an entry using this format: S# P# quantity. (enter 'exit' when done)\n");
    fgets(entryInput, 1024, stdin);
    printf("Attemping to create Entry from:: -%s-\n", entryInput);

    if( strcmp(entryInput, "exit") == 0){
      printf("Exitting\n");
      break;
    }
    char *supplierID = strtok(entryInput, " \n");
printf("user input ID: %s\n", supplierID);
    char *partID = strtok(NULL, " \n");
printf("user input part ID: %s\n");
    int quantity = atoi(strtok(NULL, " \n"));
printf("user input quantity: %d\n", quantity);

    strcpy(statement, "INSERT INTO sp VALUES (:v1, :v2, :v3)");
    exec sql prepare s from :statement;
    if (sqlca.sqlcode < 0){
      printf("Statement not prepared!\n");
      break;
    }

    exec sql execute s using :supplierID, :partID, :quantity;
    if (sqlca.sqlcode < 0){
      printf("Population not complete!\n");
    }
    else {
      printf("***Population completed***\n");
    }
  }
}


int main(){
  char userInput[100];
  exec sql connect :identity;
  if (sqlca.sqlcode < 0){
    printf("Could not connect to oracle!\n");
    exit(1);
  }
  else{
    printf("Connected to Oracle\n");
  }

  exec sql set transaction read write;
  while (1){
    printf("Let's add some table entries!\n\nEnter table name to add to (or 'exit'):");
    printf(" ");
    fgets(userInput, 100, stdin);
    //set the last character to null, instead of a newline character
    userInput[strlen(userInput) - 1] = 0;

    printf("\nValue entered was %s...\n", userInput);

    //If the user enters 'quit' then quit the program and commit changes
    if (strcmp(userInput, "exit") == 0){
      printf("Exitting\n");
      break;
    }

    if ( strcmp(userInput, "Supplier") == 0){
      populateSupplierTable();
    }
    else if ( strcmp(userInput, "Part") == 0){
      populatePartTable();
    }
    else if ( strcmp(userInput, "SP") == 0){
      populateSPTable();
    }
  }
exec sql commit release;
exit(0);
}
```



```
[fedora@OracleVM ~]$ ./p1q2
Connected to Oracle
Let's add some table entries!

Enter table name to add to (or 'exit'): Supplier

Value entered was Supplier...
Enter a supplier using this format:  S# name status city. (enter 'exit' when done):
 S2 Jones 30 Paris
Attemping to create Supplier: -S2 Jones 30 Paris

Supplier ID: S2
Supplier name: Jones
Supplier status: 30
Supplier city: Paris
***Population completed***
Enter a supplier using this format:  S# name status city. (enter 'exit' when done):
 S3 Blake 30 Paris
Attemping to create Supplier: -S3 Blake 30 Paris

Supplier ID: S3
Supplier name: Blake
Supplier status: 30
Supplier city: Paris
***Population completed***
Enter a supplier using this format:  S# name status city. (enter 'exit' when done):
 S4 Orhun 20 London
Attemping to create Supplier: -S4 Orhun 20 London

Supplier ID: S4
Supplier name: Orhun
Supplier status: 20
Supplier city: London
***Population completed***
Enter a supplier using this format:  S# name status city. (enter 'exit' when done):
 S5 Adams 30 Athens
Attemping to create Supplier: -S5 Adams 30 Athens

Supplier ID: S5
Supplier name: Adams
Supplier status: 30
Supplier city: Athens
***Population completed***
```



```
SQL> select * from supplier;

S# SNAME          STATUS CITY
-- ---------- ---------- ----------
S1 Smith              20 London
S2 Jones              30 Paris
S3 Blake              30 Paris
S4 Orhun              20 London
S5 Adams              30 Athens
```

3.

```
#include <stdio.h>
exec sql include sqlca;
exec sql begin declare section;
  char sqlstmt[1024];
  char sno[4];
  char sname[11];
  int  status;
  char city[10];
  char pno[4];
  char pname[10];
  char color[6];
  float weight;
  int qty;
  int  value = 1000;
  char *username= "fedora";
  char *password= "oracle";
exec sql end declare section;
int main() {
/*Connect DB */
exec sql connect :username identified by :password;
if (sqlca.sqlcode == 0)
    printf("Oracle connected\n");
else
     printf("Oracle not connected\n");
exec sql declare a_cursor cursor for
        select *
         from   supplier
         order by s#;
exec sql open a_cursor;
if (sqlca.sqlcode == 0)
    printf("Supplier Cursor hath been opened\n");
else
    printf("Supplier Cursor error\n");
printf("s# sname        status  city\n");
printf("-- ------------ ------  --------\n");
exec sql fetch a_cursor into :sno, :sname, :status, :city;
while(sqlca.sqlcode==0) {
    printf("%2s  %11s %4d %10s \n", sno, sname, status, city);
    strcpy(sqlstmt, "select p.p#,  pname, color, weight, p.city from supplier s, part p, sp where s.s# = sp.s# and p.p# = sp.p# and
exec sql prepare t from :sqlstmt;
if (sqlca.sqlcode == 0) {
    exec sql declare b_cursor cursor for t;
    exec sql open b_cursor using :sno;
    if (sqlca.sqlcode == 0) {
        printf("p# pname        color   weight iity\n");
        printf("-- ------------ ------- ------ --------\n");
        exec sql fetch b_cursor into :pno, :pname, :color, :weight, :city;
        while(sqlca.sqlcode==0) {
            printf("%2s %11s %10s %2.1f %10s \n", pno, pname, color, weight, city);
            exec sql fetch b_cursor into :pno, :pname, :color, :weight, :city;
        }
        }
    else {
        printf("Part Cursor error\n");
    }
    printf("---------------------------------------- \n");
    exec sql close b_cursor;
  }
  else {
      printf("statement not prepared \n");
  }
  exec sql fetch a_cursor into :sno, :sname, :status, :city;
  }
  printf("---------- \n");
  exec sql close a_cursor;
  exec sql commit release;
  exit(0);
}
```