Samuel Diamantstein(101060342): Assignment 6

PART 1

```
#include <stdio.h>
exec sql include sqlca;
exec sql begin declare section;
char sqlstmt1[1024];
char sqlstmt2[1024];
char sqlstmt3[1024];
  char *MYID= "fedora/thisvmsucks";
exec sql end declare section;

int main(){
  exec sql connect :MYID;
  if (sqlca.sqlcode == 0)
      printf("Connected to ORACLE\n");
  else
      printf("Connect Failed\n");

/*  create table account using dynamic methid 1         */
        printf("Creating tables:\n");
        strcpy(sqlstmt1,
        "create table Bank(B# varchar(4) NOT NULL, Name varchar(20) NOT NULL,
         ↪  City varchar(20) NOT NULL, primary key (B#))" );
        strcpy(sqlstmt2,
        "create table Customer(C# varchar(4) NOT NULL, Name varchar(20) NOT NULL,
         ↪  Age int NOT NULL, City varchar(10) NOT NULL,  primary key (C#))" );
        strcpy(sqlstmt3,
        "create table Account (C# varchar(4) NOT NULL, B# varchar(4) NOT NULL,
         ↪  Balance int NOT NULL, primary key (C#,B#), foreign key (C#)
         ↪  references Customer(C#) ON DELETE CASCADE, foreign key (B#)
         ↪  references Bank(B#) ON DELETE CASCADE)");

  exec sql set transaction read write;
  exec sql execute immediate :sqlstmt1;
        exec sql execute immediate :sqlstmt2;
```

```
29          exec sql execute immediate :sqlstmt3;
30  printf("Finished creating tables\n");
31
32  finish:
33  printf("Creation all done\n");
34  exit(0);
35
36    if (sqlca.sqlcode == 0)
37        printf("Table created \n");
38    else
39        printf("Table not created\n");
40
41  }
```

```
[fedora@OracleVM ~]$ ./a6part1
Connected to ORACLE
Creating tables:
Finished creating tables
Creation all done
[fedora@OracleVM ~]$
```

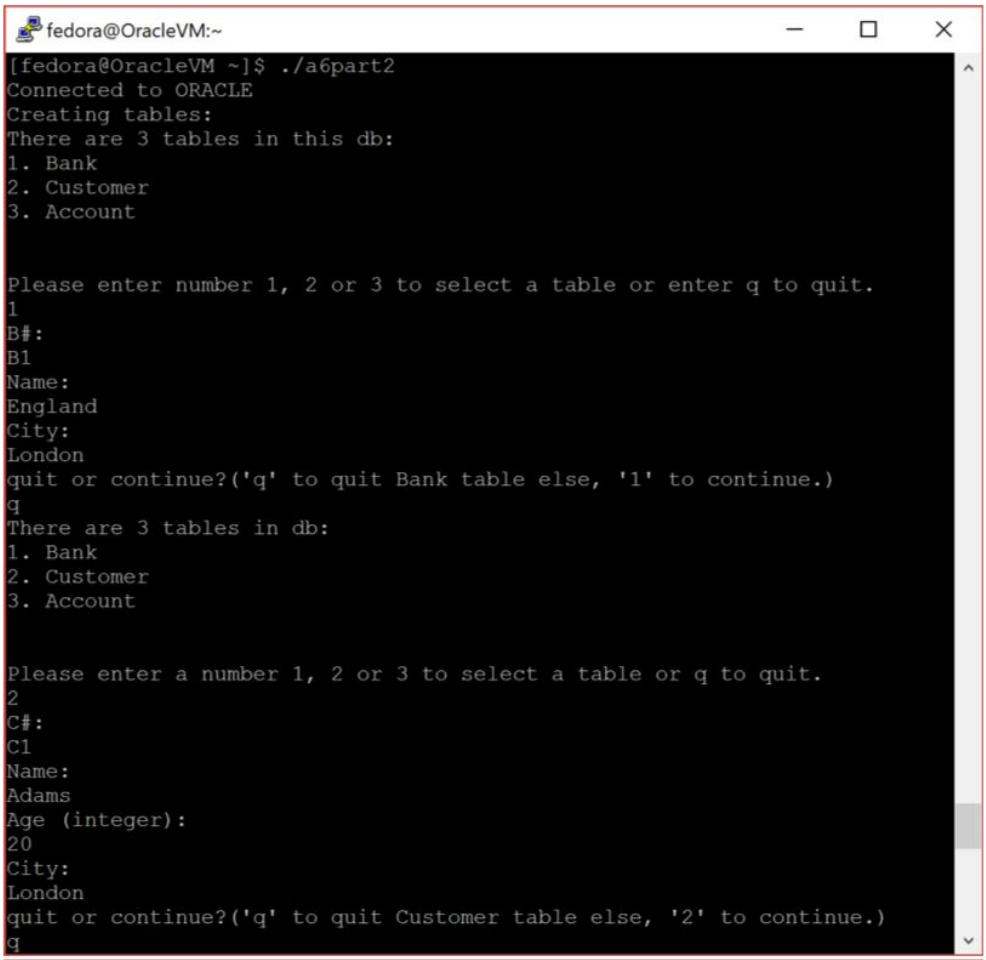
PART 2

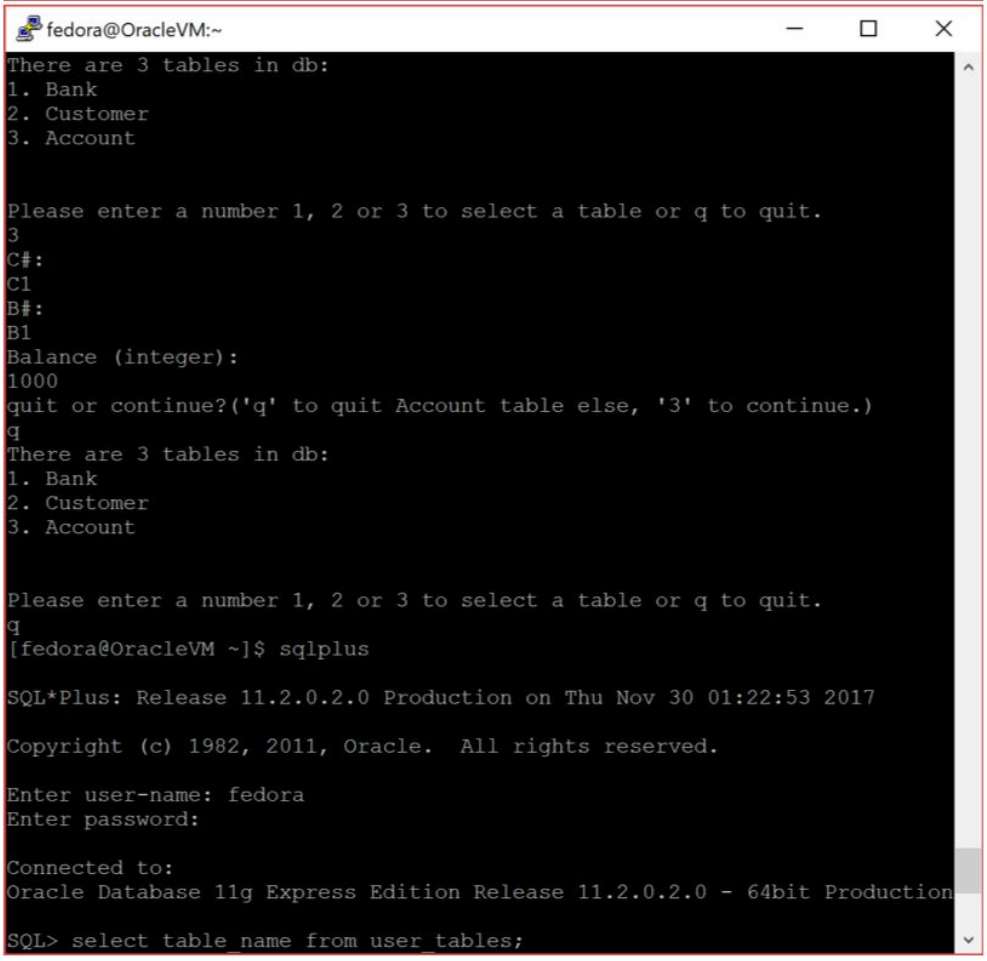
```
1  #include <stdio.h>
2  exec sql include sqlca;
3  exec sql begin declare section;
4  char bnum[5];
5  char bname[20];
6  int age;
```

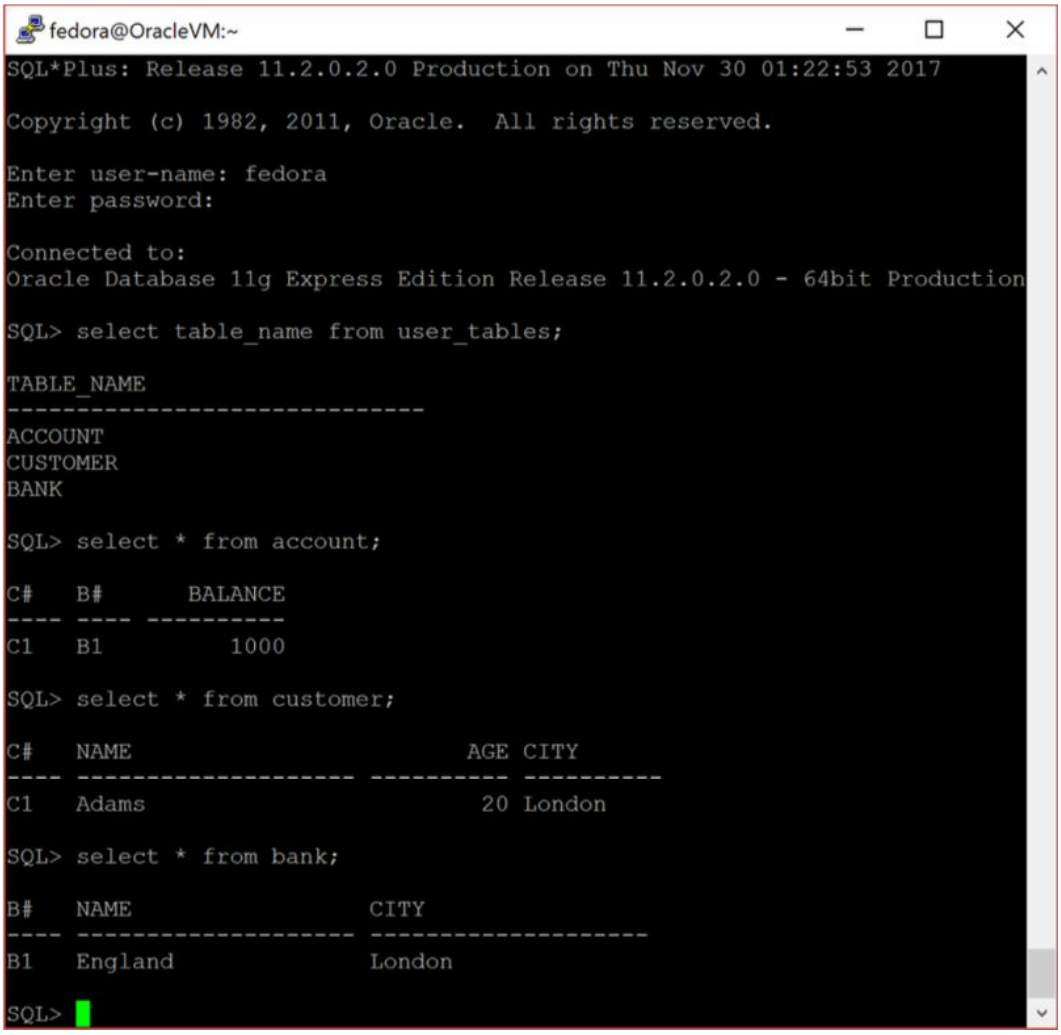
```
char bcity[20];
char cnum[5];
char cname[10];
char ccity[10];
int balance;
char sqlstmt[1024];
char pick[2];
char *MYID = "fedora/thisvmsucks";
exec sql end declare section;
int main() {
  exec sql connect : MYID;
  if (sqlca.sqlcode == 0)
    printf("Connected to ORACLE\n");
  else
    printf("Connect Failed\n");
  printf("Creating tables:\n");
  strcpy(sqlstmt,
         "create table Bank(B# varchar(4) NOT NULL, Name varchar(20) NOT NULL, "
         "City varchar(20) NOT NULL, primary key (B#))");
  exec sql set transaction read write;
  exec sql execute immediate : sqlstmt;
  strcpy(sqlstmt,
         "create table Customer(C# varchar(4) NOT NULL, Name varchar(20) NOT "
         "NULL,  Age int NOT NULL, City varchar(10) NOT NULL,  primary "
         "key (C#))");
  exec sql execute immediate : sqlstmt;
  strcpy(sqlstmt,
         "create table Account (C# varchar(4) NOT NULL, B# varchar(4) NOT "
         "NULL, Balance int NOT NULL, primary key (C#,B#), foreign key (C#) "
         "references Customer(C#) ON DELETE CASCADE, foreign key (B#) "
         "references Bank(B#) ON DELETE CASCADE)");
  exec sql execute immediate : sqlstmt;
      printf("There are 3 tables in this db: \n");
      printf("1. Bank \n");
      printf("2. Account \n");
      printf("3. Customer \n");
      printf("\n\n");
      printf("Please enter number 1, 2 or 3 to select a table or enter q to quit.
       ↪  \n");
      scanf("%s", &pick);
      while (strcmp(pick, "q") != 0) {
```

```
47          if (strcmp(pick, "1") == 0) {
48              while (strcmp(pick, "q") != 0) {
49                  printf("B#:\n");
50                  scanf("%s", &bnum);
51                  printf("Name:\n");
52                  scanf("%s", &bname);
53                  printf("City:\n");
54                  scanf("%s", &bcity);
55                  strcpy(sqlstmt, "insert into Bank values (:v1, :v2, :v3)");
56                  exec sql prepare s from : sqlstmt;
57                  exec sql execute s using : bnum, : bname, : bcity;
58                  printf("quit or continue?('q' to quit Bank table else, '1' to
                     ↪  continue.)\n");
59                  scanf("%s", &pick);
60              }
61          }
62          if (strcmp(pick, "2") == 0) {
63              while (strcmp(pick, "q") != 0) {
64                  printf("C#:\n");
65                  scanf("%s", &cnum);
66                  printf("Name:\n");
67                  scanf("%s", &cname);
68                  printf("Age (integer):\n");
69                  scanf("%d", &age);
70                  printf("City:\n");
71                  scanf("%s", &ccity);
72                  strcpy(sqlstmt, "insert into Customer values (:v1, :v2, :v3,
                     ↪  v4)");
73                  exec sql prepare s from : sqlstmt;
74                  exec sql execute s using : cnum, : cname, : age, : ccity;
75                  printf("quit or continue?('q' to quit Customer table else, '2'
                     ↪  to continue.)\n");
76                  scanf("%s", &pick);
77              }
78          }
79          if (strcmp(pick, "3") == 0) {
80              while (strcmp(pick, "q") != 0) {
81                  printf("C#:\n");
82                  scanf("%s", &cnum);
83                  printf("B#:\n");
84                  scanf("%s", &bnum);
```

```
85                      printf("Balance (integer):\n");
86                      scanf("%d", &balance);
87                      strcpy(sqlstmt, "insert into Account values (:v1, :v2, :v3)");
88                      exec sql prepare s from : sqlstmt;
89                      exec sql execute s using : bnum, : cnum, : balance;
90                      printf("quit or continue?('q' to quit Account table else, '3'
                         ↪  to continue.)\n");
91                      scanf("%s", &pick);
92                  }
93              }
94              printf("There are 3 tables in db: \n");
95              printf("1. Bank \n");
96              printf("2. Customer \n");
97              printf("3. Account \n");
98              printf("\n\n");
99              printf("Please enter a number 1, 2 or 3 to select a table or q to
                 ↪  quit.\n");
100             scanf("%s", &pick);
101             exec sql execute immediate "commit";
102         }
103     exit(1);
104 }
```


fedora@OracleVM:~
[fedora@OracleVM ~]$ ./a6part2
Connected to ORACLE
Creating tables:
There are 3 tables in this db:
1. Bank
2. Customer
3. Account
Please enter number 1, 2 or 3 to select a table or enter q to quit.
1
B#:
B1
Name:
England
City:
London
quit or continue?('q' to quit Bank table else, '1' to continue.)
q
There are 3 tables in db:
1. Bank
2. Customer
3. Account
Please enter a number 1, 2 or 3 to select a table or q to quit.
2
C#:
C1
Name:
Adams
Age (integer):
20
City:
London
quit or continue?('q' to quit Customer table else, '2' to continue.)
q


fedora@OracleVM:~
There are 3 tables in db:
1. Bank
2. Customer
3. Account
Please enter a number 1, 2 or 3 to select a table or q to quit.
3
C#:
C1
B#:
B1
Balance (integer):
1000
quit or continue?('q' to quit Account table else, '3' to continue.)
q
There are 3 tables in db:
1. Bank
2. Customer
3. Account
Please enter a number 1, 2 or 3 to select a table or q to quit.
q
[fedora@OracleVM ~]$ sqlplus
SQL*Plus: Release 11.2.0.2.0 Production on Thu Nov 30 01:22:53 2017
Copyright (c) 1982, 2011, Oracle. All rights reserved.
Enter user-name: fedora
Enter password:
Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
SQL> select table_name from user_tables;

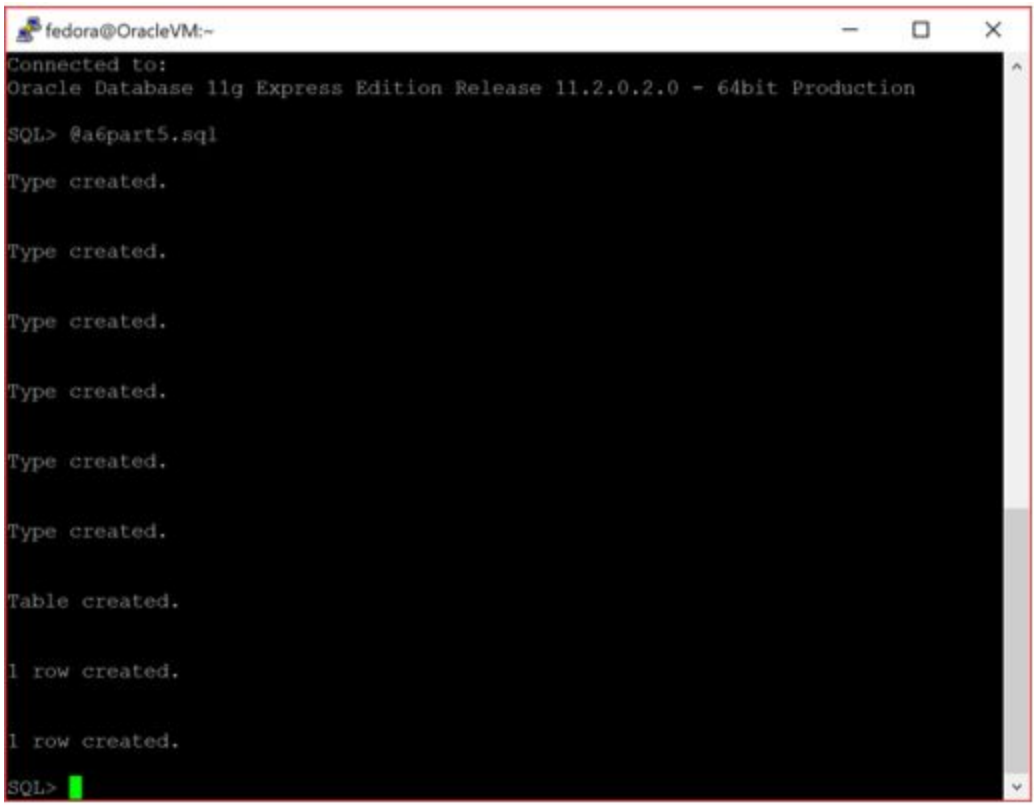


PART 3

```c
#include <stdio.h>

exec sql include sqlca;
exec sql begin declare section;
char sqlstmt1[1024];
char sqlstmt2[1024];
char bnum[3];
char bname[10];
int age;
char bcity[10];
char cnum[3];
char cname[10];
char ccity[10];
int balance;
char *MYID = "fedora/thisvmsucks";
```

```
exec sql end declare section;
int main() {
  exec sql connect : MYID;
  if (sqlca.sqlcode == 0)
    printf("Connected to ORACLE\n");
  else
    printf("Connect Failed\n");
    exec sql whenever sqlerror goto error;
    exec sql whenever not found goto done;

    strcpy(sqlstmt1, "select * from Customer order by C#");
        printf("%s", sqlstmt1);
    exec sql declare c_cur cursor for :sqlstmt1;

        strcpy(sqlstmt2, "select * from Bank where exists (select * from Account
         ↪  where Account.C#=:cnum and Account.B#=Bank.B#) order by Bank.B#");
        printf("%s", sqlstmt2);
        exec sql prepare b from :sqlstmt2;
        exec sql declare b_cur cursor for b;


  printf("         C#                Name                        Age
   ↪  City        \n");

  while(1) {
        exec sql fetch c_cur into :cnum, :cname, :age, :ccity;
        printf(" %s   %s   %d   %s \n", cnum, cname, age, ccity);

        exec sql open b_cur using :cnum;
        printf("  C#          Name     Balance        \n");
        while(1){
                exec sql fetch b_cur into :bnum, :bname, :bcity;
                printf("  %s  %s  %s \n", bnum, bname, bcity);

        }

  done:
    exec sql close c_cur;
        exec sql close b_cur;
    exec sql commit release;
    printf("Work is done successfully!\n");
```

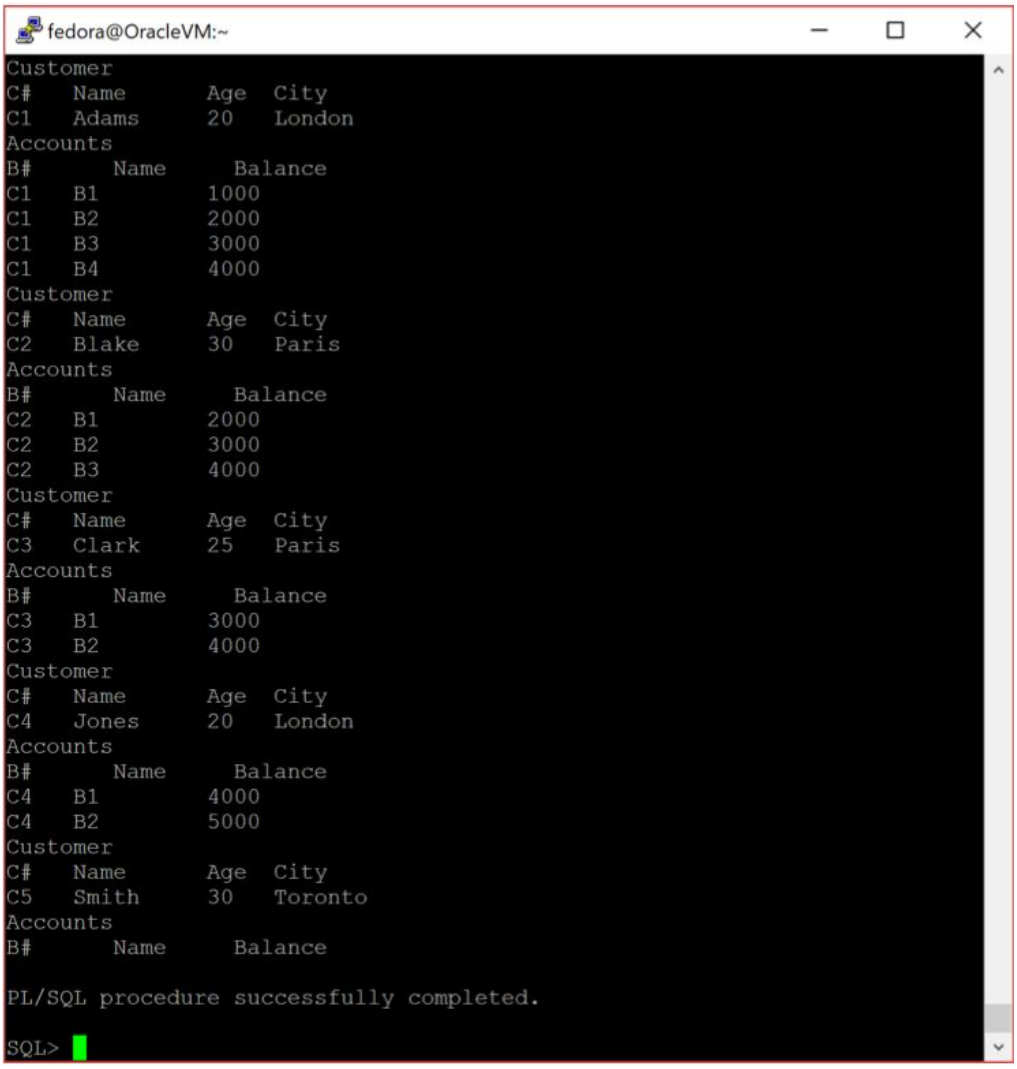
```
55      exit(0);
56
57          error:
58          printf("Error found in SQL. \n");
59          EXEC SQL WHENEVER SQLERROR CONTINUE;
60          fprintf (stderr, "DBMS Error: %.*s\n", sqlca.sqlerrm.sqlerrml,
             ↪  sqlca.sqlerrm.sqlerrmc);
61      exec sql rollback release;
62          exec sql rollback release;
63          exit(1);
64  }
```



PART 4

```
1  DECLARE
2    c_tuple Customer%rowtype;
3    b_tuple Bank%rowtype;
4
5    CURSOR c_cursor IS select * from Customer ORDER BY C# ASC;
```

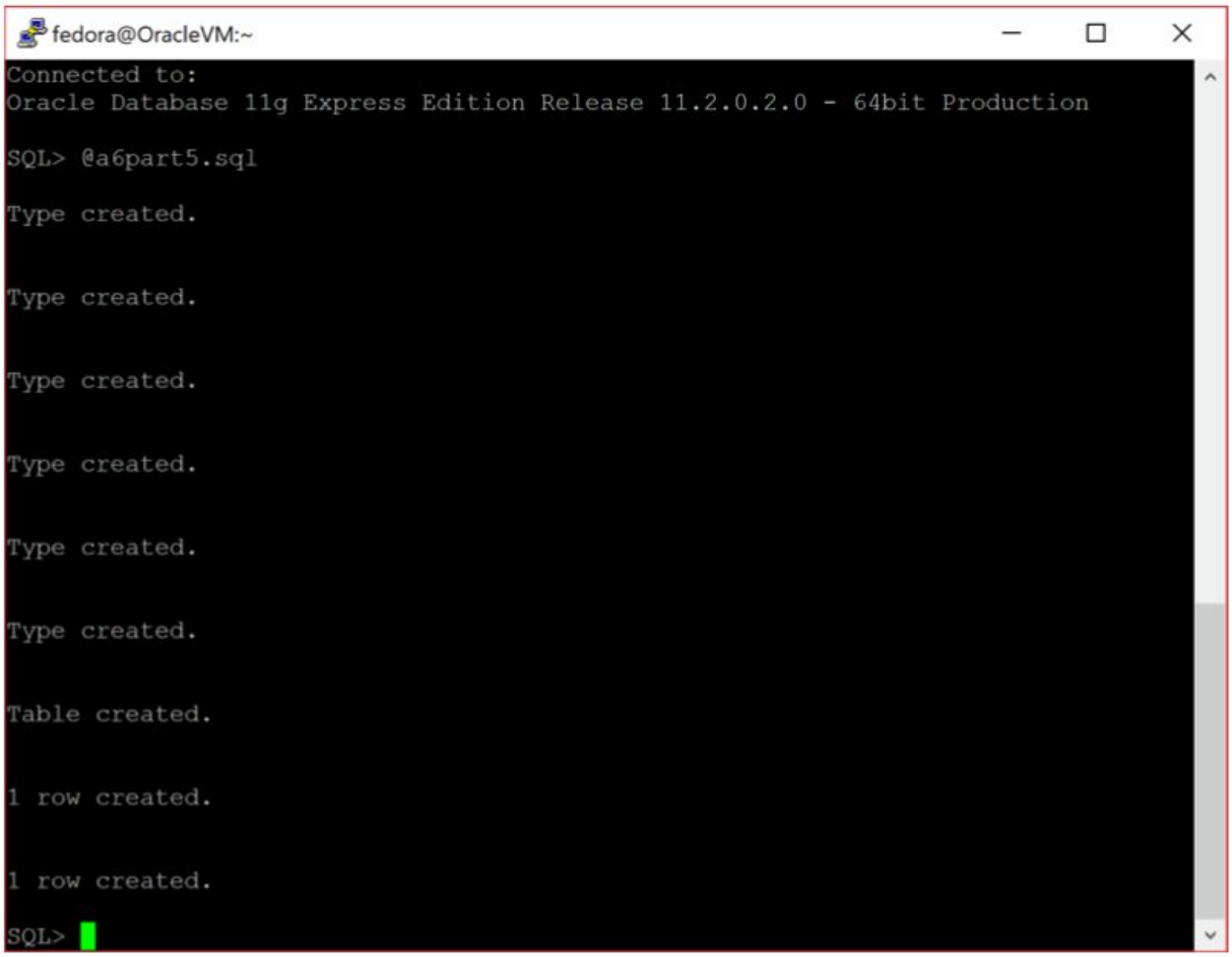
```
6     CURSOR b_cursor(cnum Customer.C#%type)
7     IS  select A.*
8         FROM Bank B, Account A
9         WHERE B.B# = A.B#
10        AND   A.C# = cnum
11        ORDER BY B.B# ASC;
12  BEGIN
13    dbms_output.put_line('  Customer');
14    dbms_output.put_line('C#   Name      Age  City      ');
15
16    OPEN c_cursor;
17    LOOP
18      fetch c_cursor into c_tuple;
19      exit when c_cursor%NOTFOUND;
20
21      dbms_output.put_line('  Customer');
22          dbms_output.put_line('C#   Name      Age  City');
23          dbms_output.put_line(rpad(to_char(c_tuple.C#),5)
24          || rpad(c_tuple.Name,10)
25          || rpad(c_tuple.Age,5)
26          || rpad(c_tuple.City,10));
27
28      dbms_output.put_line('    Accounts');
29      dbms_output.put_line('  B#      Name     Balance');
30
31      OPEN b_cursor(c_tuple.C#);
32      LOOP
33        FETCH b_cursor into b_tuple;
34        exit when b_cursor%NOTFOUND;
35
36        dbms_output.put_line(rpad(to_char(b_tuple.B#),5) ||
37          rpad(b_tuple.Name,10) || rpad(b_tuple.City,10));
38
39      END LOOP;
40      CLOSE b_cursor;
41
42    END LOOP;
43    CLOSE c_cursor;
44  END;
```



PART 5

```
1  -- types declared
2  create type name_v as varray(5) of varchar(10);
3  /
4  create type hobby_v as varray(3) of varchar(10);
5  /
6  create type student_t
7    as object(
8              name varchar(10),
9              hobby hobby_v);
10 /
11 create type student_v
12   as varray(5)
13   of student_t;
```

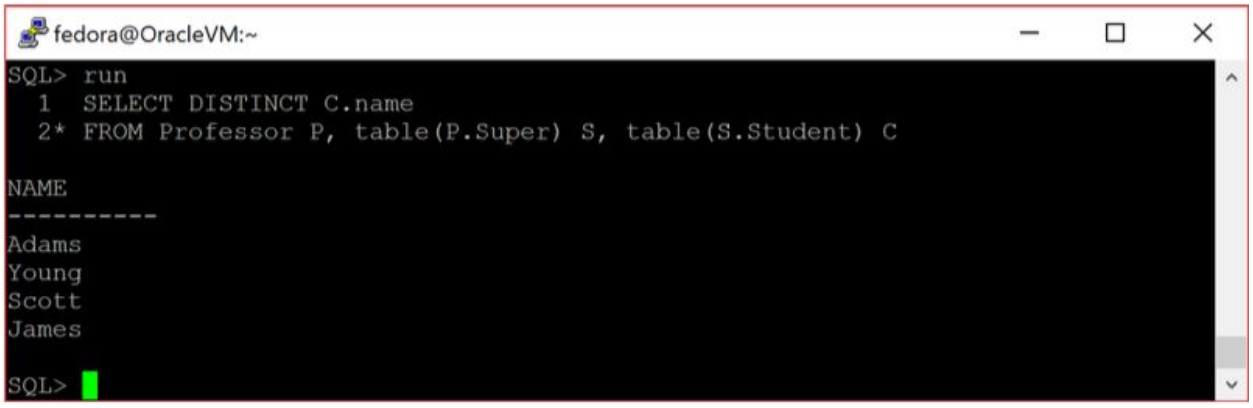
```
/
create type super_t
  as object(
            kind varchar(10),
            student student_v);
/
create type super_v as varray(5) of super_t;
/

-- table declared
create table professor(
  name varchar(10) primary key,
  hobby hobby_v,
  super super_v);

-- insert rows
insert into professor values(
  'Henry',
  hobby_v('Chess','Skiing'),
  super_v(super_t('Ph.d',
          student_v(student_t('Young',hobby_v('Skiing','Soccer')))),
  super_t('M.sc',student_v(student_t('James',hobby_v('Boxing')),
          student_t('Adams',hobby_v('Chess','Skiing'))))));
insert into professor values(
  'David',
  hobby_v('Hiking','Travel'),
  super_v(super_t('M.sc',
  student_v(student_t('Scott',hobby_v('Hiking','Travel'))))));
```



Part 6

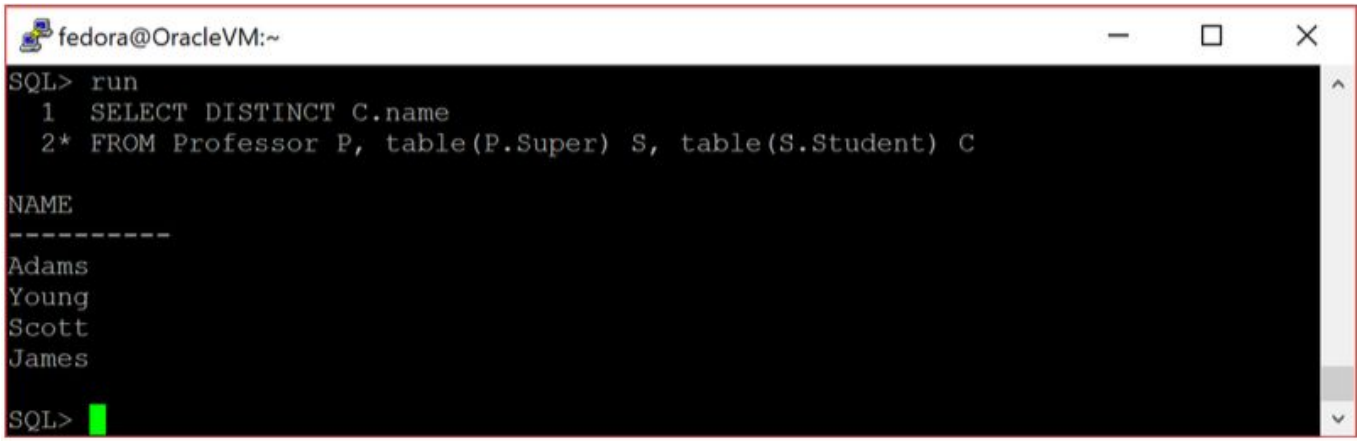
#1

```
1  -- List all students (both master and PhD) as a single set of names.
2  SELECT DISTINCT C.name
3  FROM Professor P, table(P.Super) S, table(S.Student) C;
```



#2: List professor and their students in a nested relation.

```
1 Select P.name, (CAST(multiset(
2       Select C.name
3       From Table(P.Super) S,
4               Table(S.Student) C)
5       AS name_V))
6*      From Professor P;
```



#3/#4 not done...