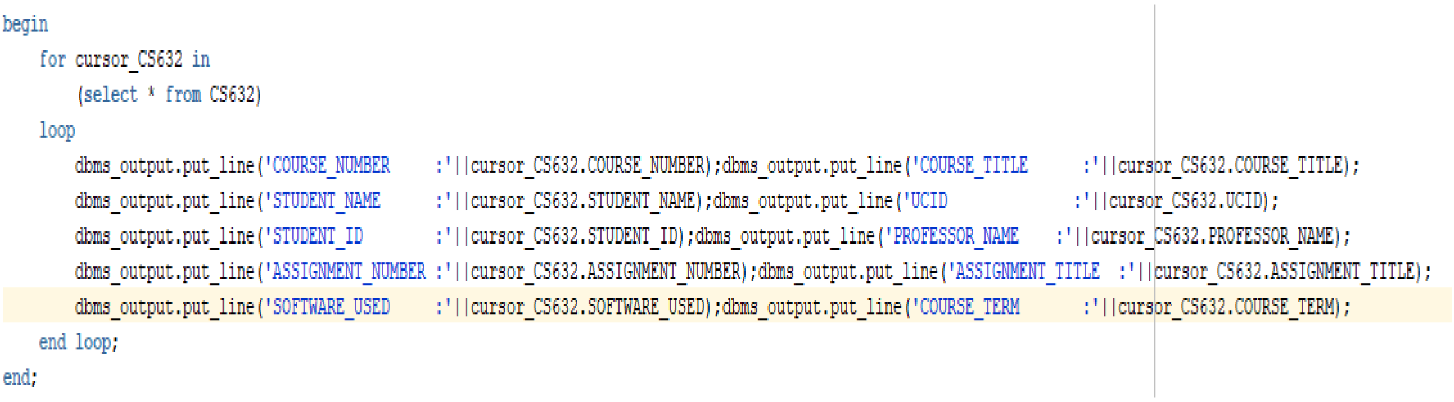
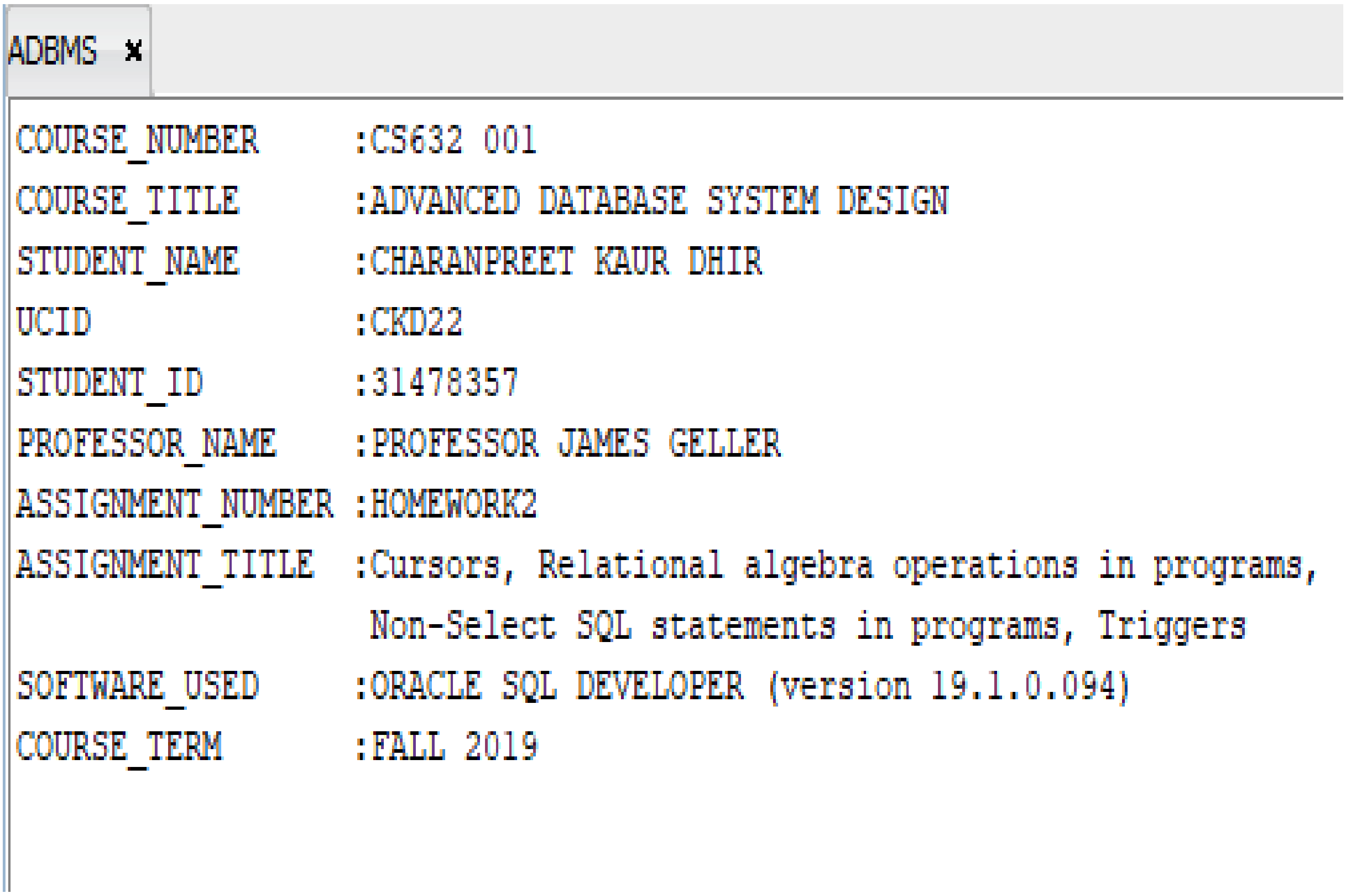
NEW JERSEY INSTITUTE OF TECHNOLGY



```
begin
    for cursor_CS632 in
        (select * from CS632)
    loop
        dbms_output.put_line('COURSE_NUMBER     :'||cursor_CS632.COURSE_NUMBER);dbms_output.put_line('COURSE_TITLE      :'||cursor_CS632.COURSE_TITLE);
        dbms_output.put_line('STUDENT_NAME      :'||cursor_CS632.STUDENT_NAME);dbms_output.put_line('UCID              :'||cursor_CS632.UCID);
        dbms_output.put_line('STUDENT_ID        :'||cursor_CS632.STUDENT_ID);dbms_output.put_line('PROFESSOR_NAME    :'||cursor_CS632.PROFESSOR_NAME);
        dbms_output.put_line('ASSIGNMENT_NUMBER :'||cursor_CS632.ASSIGNMENT_NUMBER);dbms_output.put_line('ASSIGNMENT_TITLE  :'||cursor_CS632.ASSIGNMENT_TITLE);
        dbms_output.put_line('SOFTWARE_USED     :'||cursor_CS632.SOFTWARE_USED);dbms_output.put_line('COURSE_TERM       :'||cursor_CS632.COURSE_TERM);
    end loop;
end;
```



```
COURSE_NUMBER     :CS632 001
COURSE_TITLE      :ADVANCED DATABASE SYSTEM DESIGN
STUDENT_NAME      :CHARANPREET KAUR DHIR
UCID              :CKD22
STUDENT_ID        :31478357
PROFESSOR_NAME    :PROFESSOR JAMES GELLER
ASSIGNMENT_NUMBER :HOMEWORK2
ASSIGNMENT_TITLE  :Cursors, Relational algebra operations in programs,
                   Non-Select SQL statements in programs, Triggers
SOFTWARE_USED     :ORACLE SQL DEVELOPER (version 19.1.0.094)
COURSE_TERM       :FALL 2019
```

HOMEWORK 2

1) a) Study the Oracle SQL operator length().

Answer a)

The SQL LENGTH function returns the number of characters in a string.

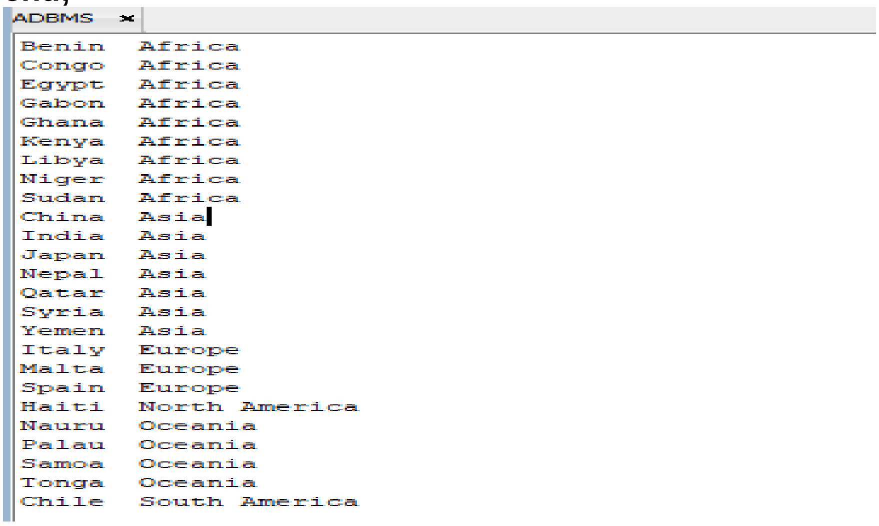
Syntax: Length(string)

b) We are reusing the table COUNTRY_CONT.
Write a PL/SQL program using an EXPLICIT cursor that will display on screen all the countries in the world that have names that are exactly 5 letters long.

All the processing has to be done in the PROGRAM. The Select statement in the cursor has to be (select * from COUNTRY-CONT). Nothing else. NO WHERE CLAUSE. Display both the country and the continent.

Answer b)

```
declare
  NewCountry varchar2(40);
  NewContinent varchar2(40);
  cursor Five_letter_Country
                    is select * from country_cont;
begin
   open Five_letter_Country;
   loop
      fetch Five_letter_Country into NewCountry,NewContinent;
      if Five_letter_Country%found
      then
         if length(NewCountry)=5
         then
            dbms_output.put_line(NewCountry||'  '||NewContinent);
         end if;
      else
         exit;
      end if;
   end loop;
   close Five_letter_Country;
end;
```

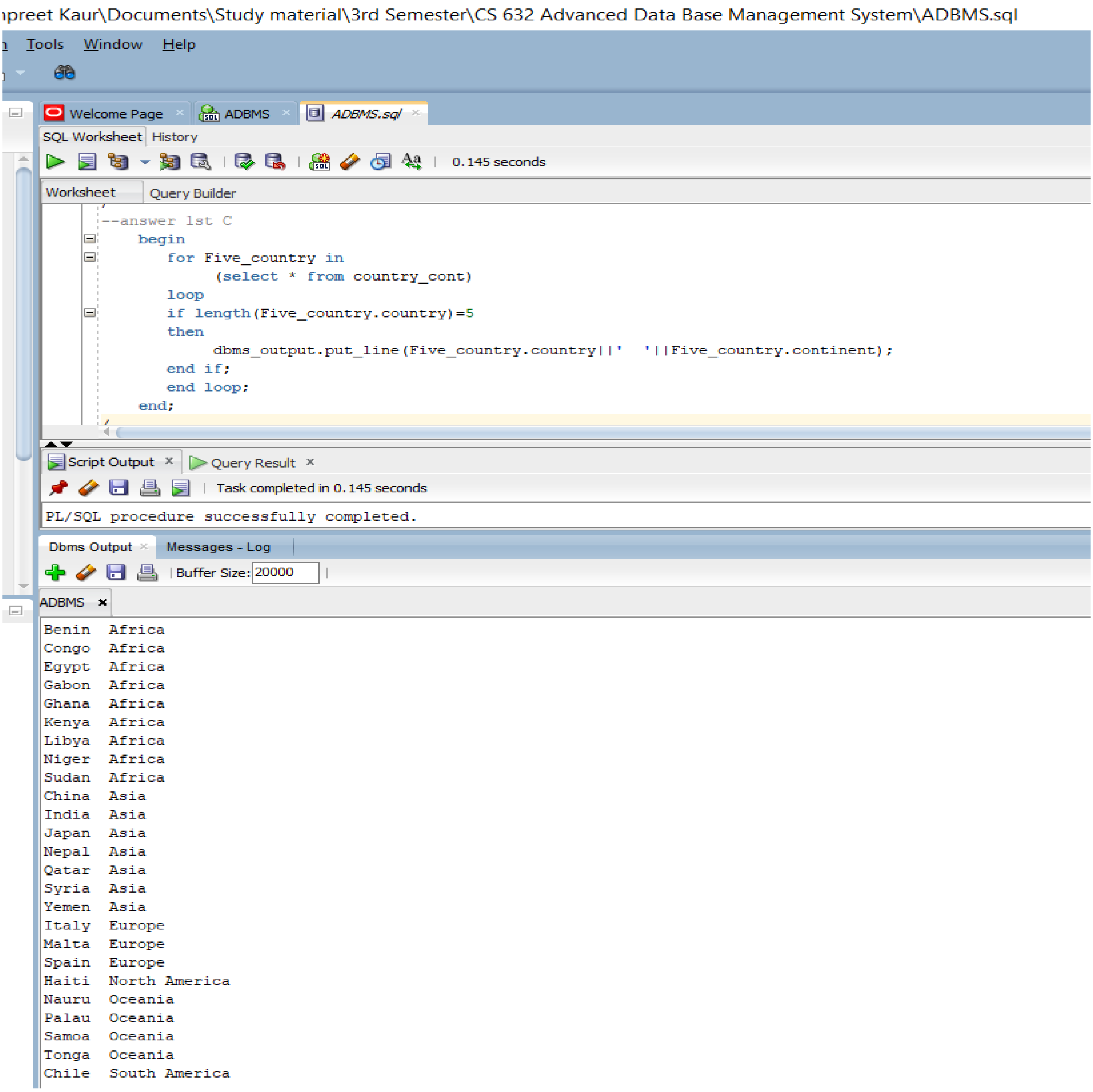


```
Benin  Africa
Congo  Africa
Egypt  Africa
Gabon  Africa
Ghana  Africa
Kenya  Africa
Libya  Africa
Niger  Africa
Sudan  Africa
China  Asia
India  Asia
Japan  Asia
Nepal  Asia
Qatar  Asia
Syria  Asia
Yemen  Asia
Italy  Europe
Malta  Europe
Spain  Europe
Haiti  North America
Nauru  Oceania
Palau  Oceania
Samoa  Oceania
Tonga  Oceania
Chile  South America
```

c) Repeat question b) with an **implicit** cursor.

Answer c)

```
begin
      for Five_country in
            (select * from country_cont)
      loop
      if length(Five_country.country)=5
      then
        dbms_output.put_line(Five_country.country||' '||Five_country.continent);
      end if;
      end loop;
          end;
```



```
--answer 1st C
    begin
       for Five_country in
            (select * from country_cont)
       loop
       if length(Five_country.country)=5
       then
            dbms_output.put_line(Five_country.country||'  '||Five_country.continent);
       end if;
       end loop;
    end;
```

```
PL/SQL procedure successfully completed.
```

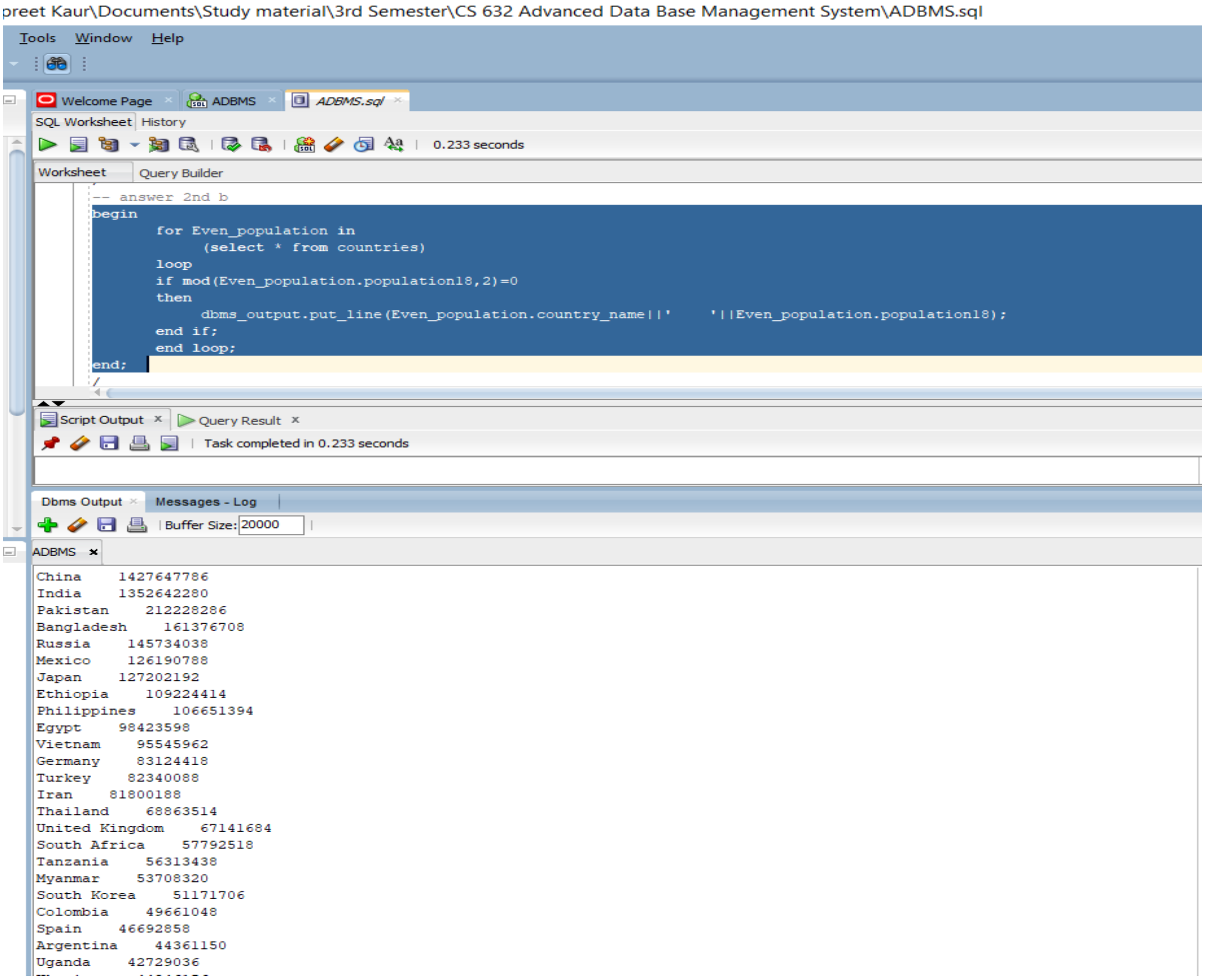
```
Benin  Africa
Congo  Africa
Egypt  Africa
Gabon  Africa
Ghana  Africa
Kenya  Africa
Libya  Africa
Niger  Africa
Sudan  Africa
China  Asia
India  Asia
Japan  Asia
Nepal  Asia
Qatar  Asia
Syria  Asia
Yemen  Asia
Italy  Europe
Malta  Europe
Spain  Europe
Haiti  North America
Nauru  Oceania
Palau  Oceania
Samoa  Oceania
Tonga  Oceania
Chile  South America
```

2) a) Study the PL/SQL operator “mod”.
Answer a)

The Oracle PL/SQL MOD (short for modulus) function returns the remainder when one argument is divided by the second argument. It accepts two number arguments and returns a single numeric value.
Syntax: MOD(x,y)

b) We are reusing the table COUNTRIES.
Write a PL/SQL program using an IMPLICT cursor that will display on screen all the countries in the world that had an EVEN number of people (population) in 2018. Show only the column of the Country name and the column of the population in 2018.
Answer b)

```
 begin
      for Even_population in
          (select * from countries)
      loop
      if mod(Even_population.population18,2)=0
      then
          dbms_output.put_line(Even_population.country_name||'
           '||Even_population.population18);
      end if;
      end loop;
end;
```



preet Kaur\Documents\Study material\3rd Semester\CS 632 Advanced Data Base Management System\ADBMS.sql

```
-- answer 2nd b
begin
       for Even_population in
            (select * from countries)
       loop
       if mod(Even_population.population18,2)=0
       then
            dbms_output.put_line(Even_population.country_name||'     '||Even_population.population18);
       end if;
       end loop;
end;
/
```

Task completed in 0.233 seconds

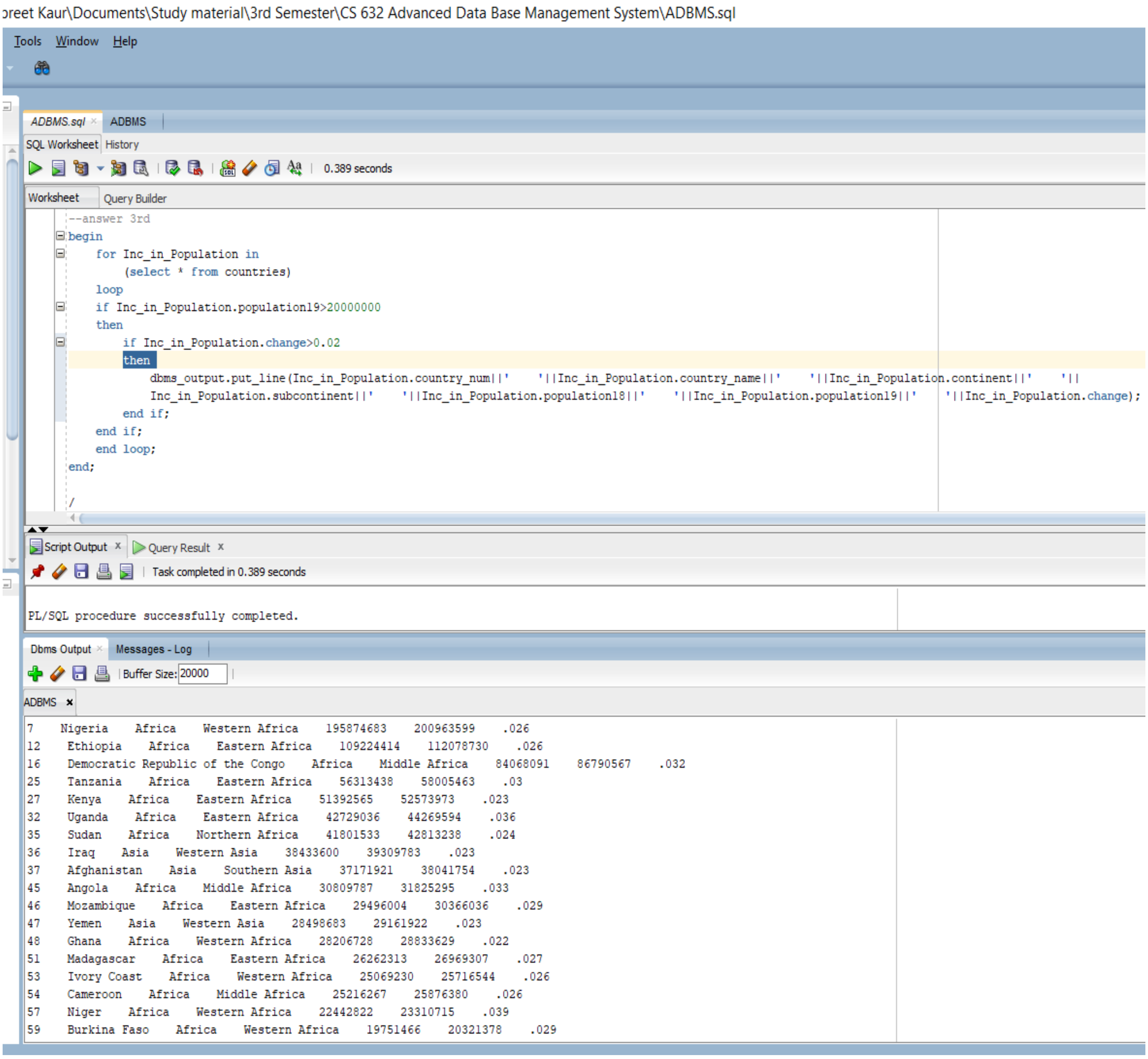
Buffer Size: 20000

```
China    1427647786
India    1352642280
Pakistan    212228286
Bangladesh    161376708
Russia    145734038
Mexico    126190788
Japan    127202192
Ethiopia    109224414
Philippines    106651394
Egypt    98423598
Vietnam    95545962
Germany    83124418
Turkey    82340088
Iran    81800188
Thailand    68863514
United Kingdom    67141684
South Africa    57792518
Tanzania    56313438
Myanmar    53708320
South Korea    51171706
Colombia    49661048
Spain    46692858
Argentina    44361150
Uganda    42729036
```

3) Write a PL/SQL program using an IMPLICT cursor that will display on screen all the countries that had a population of over 20 million in 2019 AND also a population increase of more than +2%. Display all the columns of the table COUNTRIES for these rows.

Answer 3)

```
begin
   for Inc_in_Population in
      (select * from countries)
   loop
   if Inc_in_Population.population19>20000000
   then
      if Inc_in_Population.change>0.02
      then
         dbms_output.put_line(Inc_in_Population.country_num||'
        '||Inc_in_Population.country_name||'   '||Inc_in_Population.continent||'
        '|Inc_in_Population.subcontinent||'   '||Inc_in_Population.population18||'
        '||Inc_in_Population.population19||'   '||Inc_in_Population.change);
      end if;
   end if;
   end loop;
end;
```



ɔreet Kaur\Documents\Study material\3rd Semester\CS 632 Advanced Data Base Management System\ADBMS.sql

```
--answer 3rd
begin
    for Inc_in_Population in
        (select * from countries)
    loop
    if Inc_in_Population.population19>20000000
    then
        if Inc_in_Population.change>0.02
        then
            dbms_output.put_line(Inc_in_Population.country_num||'    '||Inc_in_Population.country_name||'    '||Inc_in_Population.continent||'    '||
            Inc_in_Population.subcontinent||'    '||Inc_in_Population.population18||'    '||Inc_in_Population.population19||'    '||Inc_in_Population.change);
        end if;
    end if;
    end loop;
end;

/
```

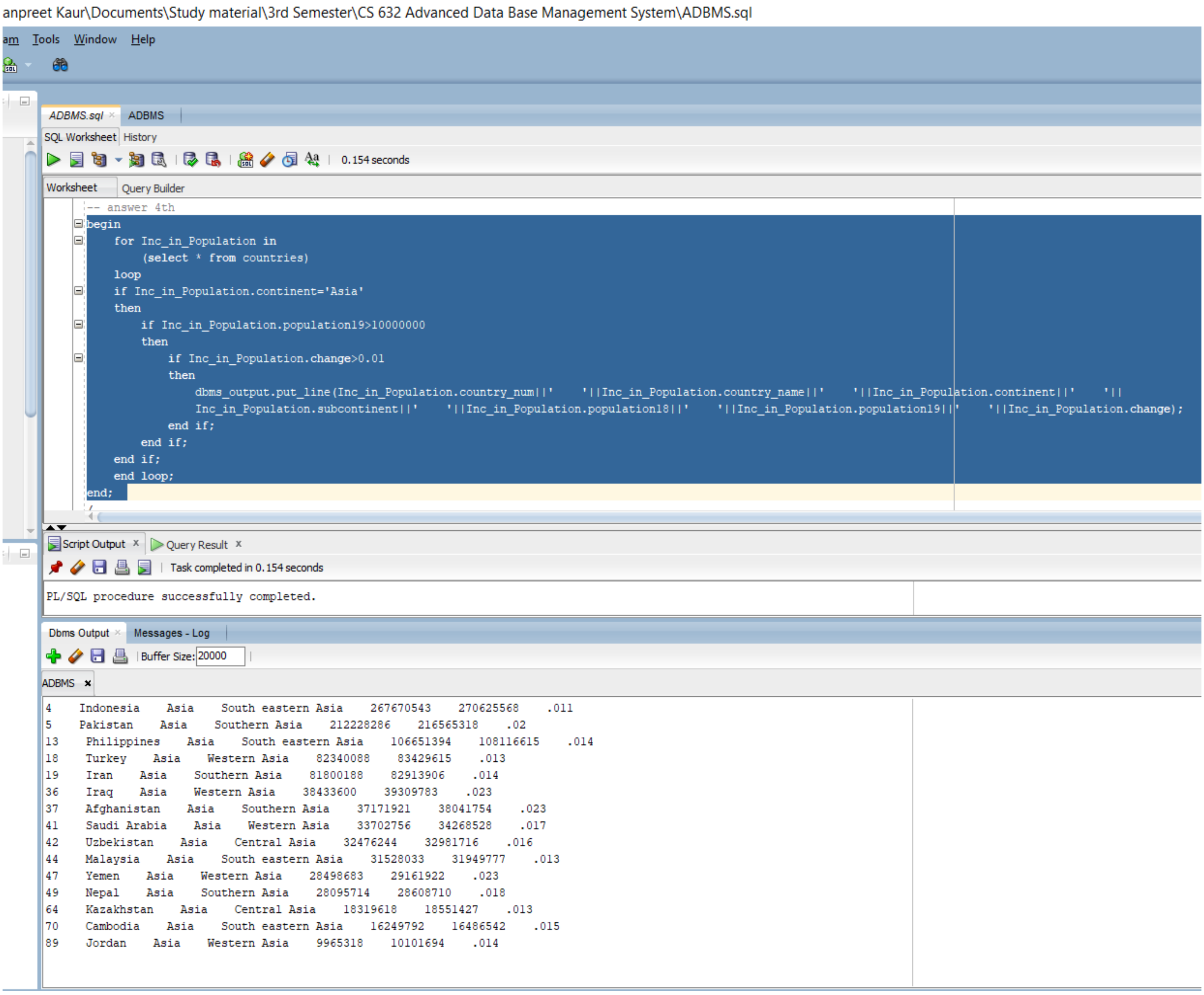
```
PL/SQL procedure successfully completed.
```

```
7   Nigeria    Africa    Western Africa    195874683    200963599    .026
12   Ethiopia    Africa    Eastern Africa    109224414    112078730    .026
16   Democratic Republic of the Congo    Africa    Middle Africa    84068091    86790567    .032
25   Tanzania    Africa    Eastern Africa    56313438    58005463    .03
27   Kenya    Africa    Eastern Africa    51392565    52573973    .023
32   Uganda    Africa    Eastern Africa    42729036    44269594    .036
35   Sudan    Africa    Northern Africa    41801533    42813238    .024
36   Iraq    Asia    Western Asia    38433600    39309783    .023
37   Afghanistan    Asia    Southern Asia    37171921    38041754    .023
45   Angola    Africa    Middle Africa    30809787    31825295    .033
46   Mozambique    Africa    Eastern Africa    29496004    30366036    .029
47   Yemen    Asia    Western Asia    28498683    29161922    .023
48   Ghana    Africa    Western Africa    28206728    28833629    .022
51   Madagascar    Africa    Eastern Africa    26262313    26969307    .027
53   Ivory Coast    Africa    Western Africa    25069230    25716544    .026
54   Cameroon    Africa    Middle Africa    25216267    25876380    .026
57   Niger    Africa    Western Africa    22442822    23310715    .039
59   Burkina Faso    Africa    Western Africa    19751466    20321378    .029
```

4) Write a PL/SQL program using an IMPLICT cursor that will display on screen all the countries IN ASIA that had a population of over 10 million in 2019 AND also a population increase of more than **+1%**. Display all the columns of the table COUNTRIES for these rows.

Answer 4)

```
begin
    for Inc_in_Population in
        (select * from countries)
    loop
    if Inc_in_Population.continent='Asia'
    then
        if Inc_in_Population.population19>10000000
        then
            if Inc_in_Population.change>0.01
            then
              dbms_output.put_line(Inc_in_Population.country_num||'
            '||Inc_in_Population.country_name||'    '||Inc_in_Population.continent||'    '||
            Inc_in_Population.subcontinent||' '||Inc_in_Population.population18||'
            '||Inc_in_Population.population19||'    '||Inc_in_Population.change);
            end if;
        end if;
    end if;
    end loop;
end;
```



anpreet Kaur\Documents\Study material\3rd Semester\CS 632 Advanced Data Base Management System\ADBMS.sql

```
-- answer 4th
begin
    for Inc_in_Population in
        (select * from countries)
    loop
    if Inc_in_Population.continent='Asia'
    then
        if Inc_in_Population.population19>10000000
        then
            if Inc_in_Population.change>0.01
            then
                dbms_output.put_line(Inc_in_Population.country_num||'    '||Inc_in_Population.country_name||'    '||Inc_in_Population.continent||'    '||
                Inc_in_Population.subcontinent||'    '||Inc_in_Population.population18||'    '||Inc_in_Population.population19||'    '||Inc_in_Population.change);
            end if;
        end if;
    end if;
    end loop;
end;
```

```
PL/SQL procedure successfully completed.
```

```
4    Indonesia    Asia    South eastern Asia    267670543    270625568    .011
5    Pakistan    Asia    Southern Asia    212228286    216565318    .02
13    Philippines    Asia    South eastern Asia    106651394    108116615    .014
18    Turkey    Asia    Western Asia    82340088    83429615    .013
19    Iran    Asia    Southern Asia    81800188    82913906    .014
36    Iraq    Asia    Western Asia    38433600    39309783    .023
37    Afghanistan    Asia    Southern Asia    37171921    38041754    .023
41    Saudi Arabia    Asia    Western Asia    33702756    34268528    .017
42    Uzbekistan    Asia    Central Asia    32476244    32981716    .016
44    Malaysia    Asia    South eastern Asia    31528033    31949777    .013
47    Yemen    Asia    Western Asia    28498683    29161922    .023
49    Nepal    Asia    Southern Asia    28095714    28608710    .018
64    Kazakhstan    Asia    Central Asia    18319618    18551427    .013
70    Cambodia    Asia    South eastern Asia    16249792    16486542    .015
89    Jordan    Asia    Western Asia    9965318    10101694    .014
```

5) a) Create a new table CONT_COUNTRY that has the columns CONTINENT and COUNTRY, i.e., in the opposite order of the columns COUNTRY-CONT.
Show the CREATE statement.

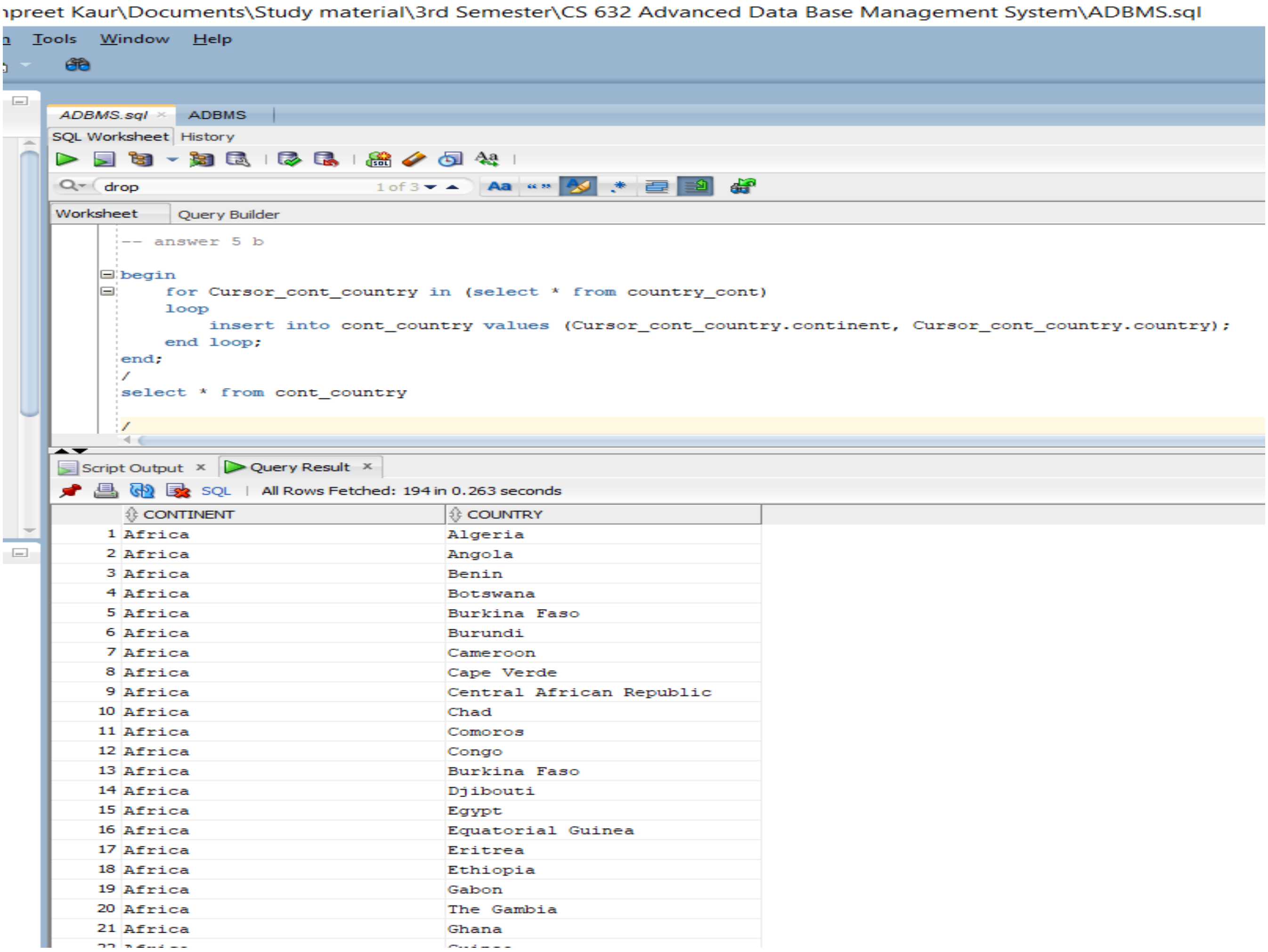
Answer a)

```
create table CONT_COUNTRY (
continent varchar(20),
country varchar(48)
)
```

b) Write a PL/SQL program using an implicit cursor on COUNTRY_CONT that will insert all data values into CONT_COUNTRY. In other words, in COUNTRY_CONT the first row is
Algeria Africa
and in CONT_COUNTRY the first row will be
Africa Algeria
Show the program and the resulting table.

Answer b)

```
begin
   for Cursor_cont_country in (select * from country_cont)
   loop
        insert into cont_country values (Cursor_cont_country.continent,
         Cursor_cont_country.country);
   end loop;
end;
/
select * from cont_country
```



```
-- answer 5 b

begin
    for Cursor_cont_country in (select * from country_cont)
    loop
        insert into cont_country values (Cursor_cont_country.continent, Cursor_cont_country.country);
    end loop;
end;
/
select * from cont_country

/
```

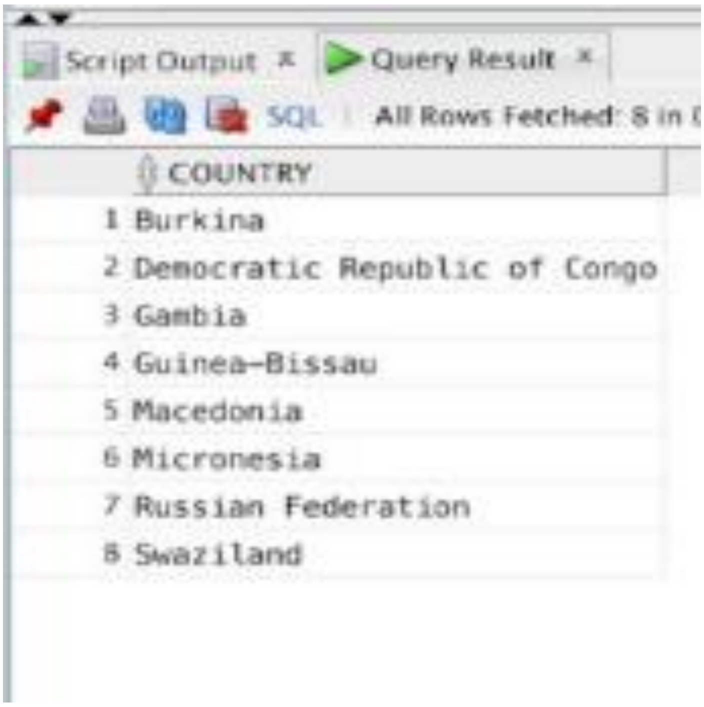
All Rows Fetched: 194 in 0.263 seconds

	CONTINENT	COUNTRY
1	Africa	Algeria
2	Africa	Angola
3	Africa	Benin
4	Africa	Botswana
5	Africa	Burkina Faso
6	Africa	Burundi
7	Africa	Cameroon
8	Africa	Cape Verde
9	Africa	Central African Republic
10	Africa	Chad
11	Africa	Comoros
12	Africa	Congo
13	Africa	Burkina Faso
14	Africa	Djibouti
15	Africa	Egypt
16	Africa	Equatorial Guinea
17	Africa	Eritrea
18	Africa	Ethiopia
19	Africa	Gabon
20	Africa	The Gambia
21	Africa	Ghana
22	Africa	Guinea

6) a) Using the SET difference operator in SQL (NOT PL/SQL) find the list of all countries that exist in COUNTRY_CONT but that are missing in COUNTRIES. If there are such countries it is likely that this is a data cleaning issue, because the table COUNTRIES appears rather complete.Show the countries you find this way, if any.

Answer a)

```
select country from Country_cont
minus
select country_name from countries
```



All Rows Fetched: 8 in (

	COUNTRY
1	Burkina
2	Democratic Republic of Congo
3	Gambia
4	Guinea-Bissau
5	Macedonia
6	Micronesia
7	Russian Federation
8	Swaziland

b) If you find any countries then “clean” the countries in the table COUNTRY_CONT so that they have the same names as in COUNTRIES.
Use SQL UPDATE statements for the cleaning. Show the UPDATE statements.

Answer b)

```
update country_cont set country='Burkina Faso' where country='Burkina'

update country_cont set country='Burkina Faso' where country like '%of Congo'

update country_cont set country='The Gambia' where country = 'Gambia'

update country_cont set country='Guinea Bissau' where country = 'GuineaBissau'

update country_cont set country='North Macedonia' where country='Macedonia'

update country_cont set country='Federated States of Micronesia' where country = 'Micronesia'

update country_cont set country='Russia' where country = 'Russian Federation'

update country_cont set country='Eswatini Swaziland' where country = 'Swaziland'
```

After Update:

c) Using the SET difference operator in SQL (NOT PL/SQL) find the list of all countries that exist in COUNTRIES but that are missing in COUNTRY_CONT. Create a new table ADDITIONAL that contains these countries in one single column called COUNTRIES. As always: Show the SQL operation. Show the resulting table.

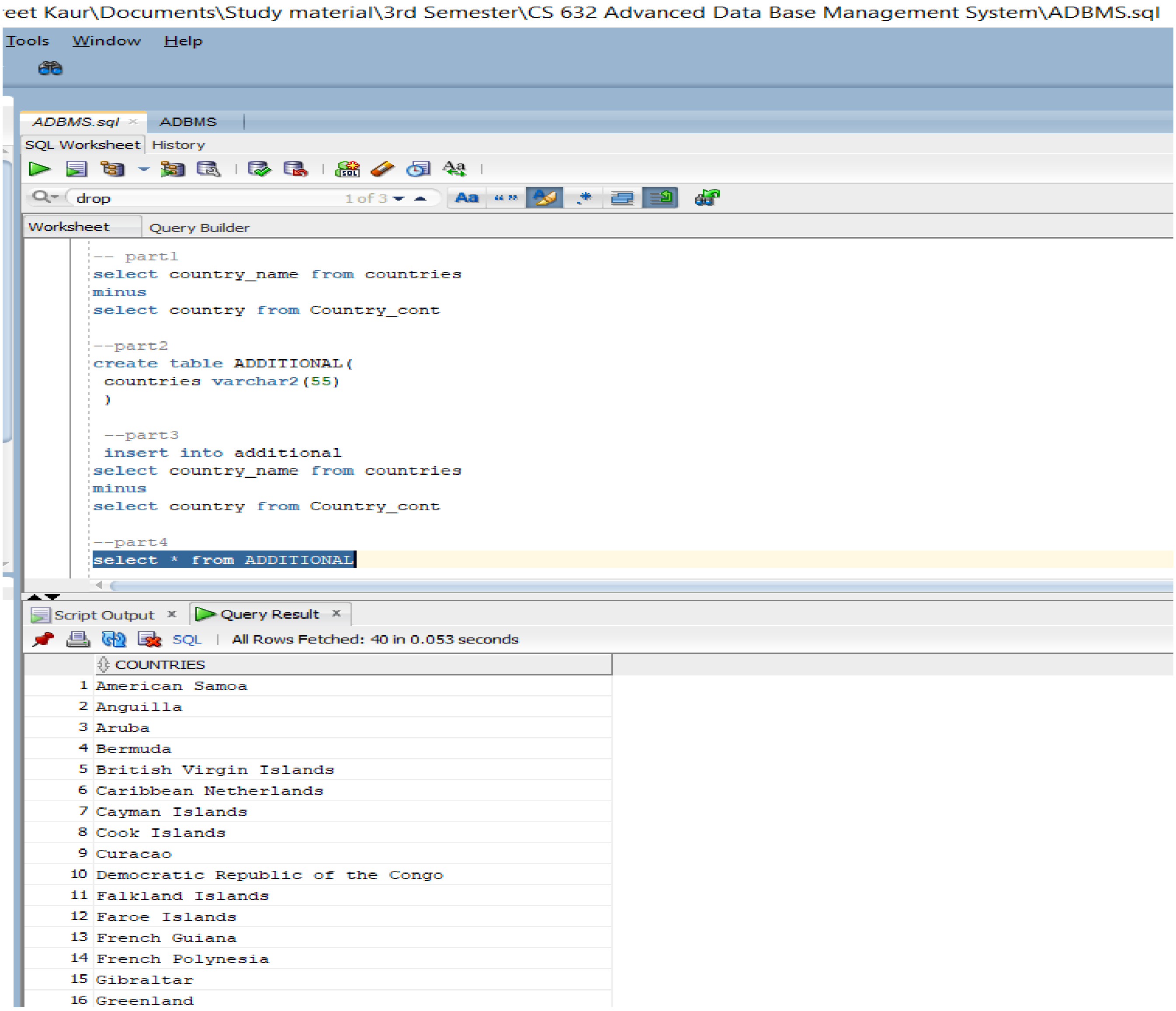
Answer c)

```
-- part1
select country_name from countries
minus
select country from Country_cont

--part2
create table ADDITIONAL(
 countries varchar2(55)
 )

 --part3
 insert into additional
select country_name from countries
minus
select country from Country_cont

--part4
select * from ADDITIONAL
```



eet Kaur\Documents\Study material\3rd Semester\CS 632 Advanced Data Base Management System\ADBMS.sql

```
-- part1
select country_name from countries
minus
select country from Country_cont

--part2
create table ADDITIONAL(
 countries varchar2(55)
 )

 --part3
 insert into additional
select country_name from countries
minus
select country from Country_cont

--part4
select * from ADDITIONAL
```

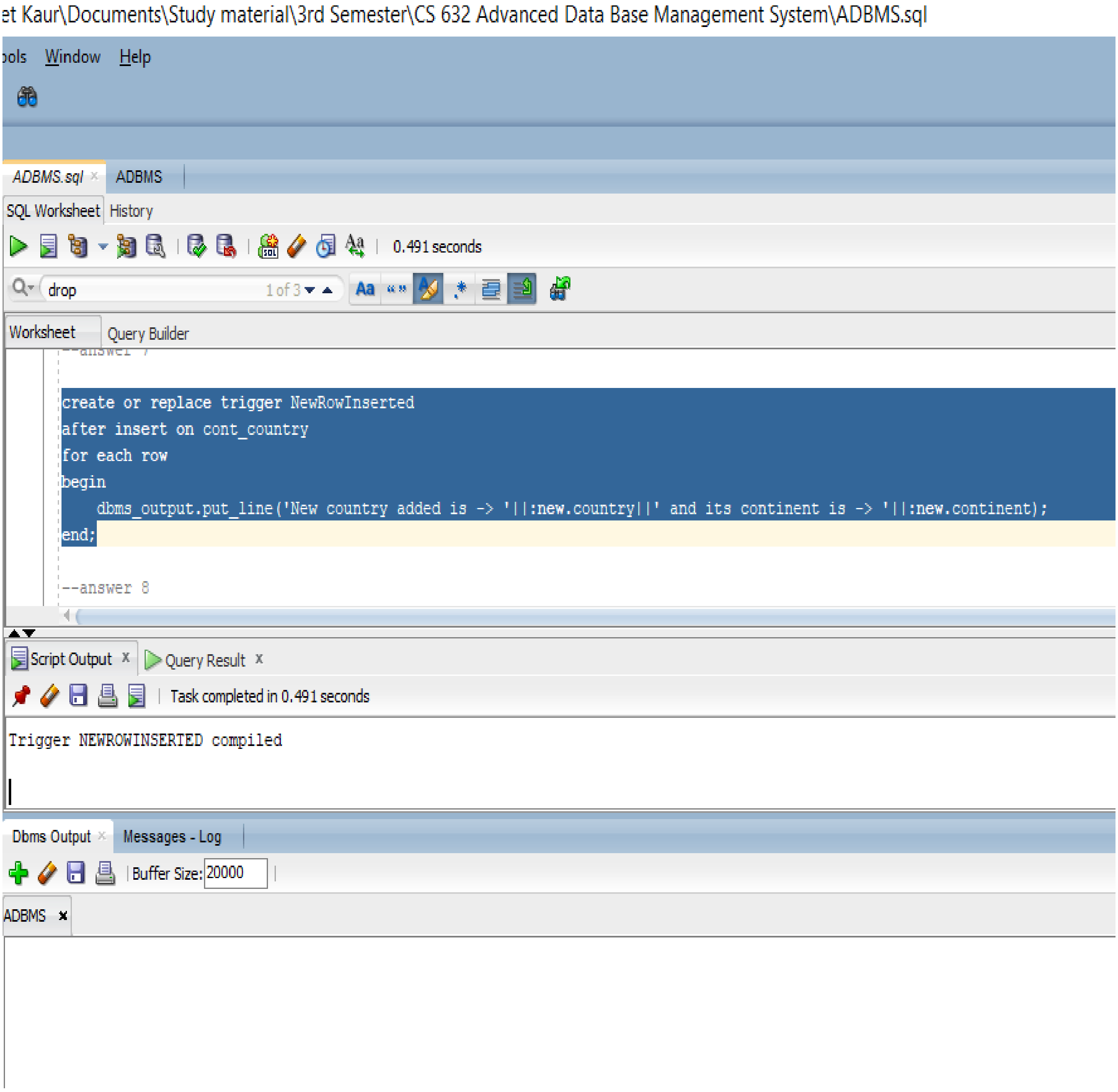
All Rows Fetched: 40 in 0.053 seconds

	COUNTRIES
1	American Samoa
2	Anguilla
3	Aruba
4	Bermuda
5	British Virgin Islands
6	Caribbean Netherlands
7	Cayman Islands
8	Cook Islands
9	Curacao
10	Democratic Republic of the Congo
11	Falkland Islands
12	Faroe Islands
13	French Guiana
14	French Polynesia
15	Gibraltar
16	Greenland

7) Write a PL/SQL insert TRIGGER on the table CONT_COUNTRY that will send a message to the screen for every inserted row with a friendly message what country and continent are being inserted.

Answer 7)

```
create or replace trigger NewRowInserted
after insert on cont_country
for each row
begin
   dbms_output.put_line('New country added is -> '||:new.country||' and its
    continent is ->  '||:new.continent);
end;
```



```
--answer 7

create or replace trigger NewRowInserted
after insert on cont_country
for each row
begin
    dbms_output.put_line('New country added is -> '||:new.country||' and its continent is -> '||:new.continent);
end;

--answer 8
```

```
Trigger NEWROWINSERTED compiled
```

8) Write a PL/SQL function CONTINENT_OF that takes one in parameter ACOUNTRY and returns the continent of that country according to the table COUNTRIES.
Run 5 trial runs of the function by passing countries *of your choice* from five different continents into the function. Write one single minimal main program to do these 5 calls. Show the function, show the main program, show the results of running the main program.

Answer 8)

```
create or replace function continent_of(acountry in varchar2)
return varchar2
as
   name_of_continent varchar2(55);
begin
   select continent into name_of_continent from countries where
   country_name=acountry;
   return name_of_continent;
end;
/

declare
   give_country varchar2(55);
   give_continent varchar2(55);
begin
   give_country := 'Chile';
   give_continent := continent_of(give_country);
   dbms_output.put_line('Continent for country '||give_country||' is -> '||
    give_continent);
   dbms_output.put_line('   ');

   give_country := 'India';
   give_continent := continent_of(give_country);
   dbms_output.put_line('Continent is '|| give_country||' is -> '|| give_continent);
   dbms_output.put_line('   ');

   give_country := 'Greece';
   give_continent := continent_of(give_country);
   dbms_output.put_line('Continent is '||give_country||' is -> '|| give_continent);
   dbms_output.put_line('   ');

   give_country := 'Egypt';
   give_continent := continent_of(give_country);
   dbms_output.put_line('Continent is '||give_country||' is -> '||give_continent);
   dbms_output.put_line('   ');

   give_country := 'El Salvador';
   give_continent := continent_of(give_country);
   dbms_output.put_line('Continent is '||give_country||' is -> '|| give_continent);
end;
```

Output file is:

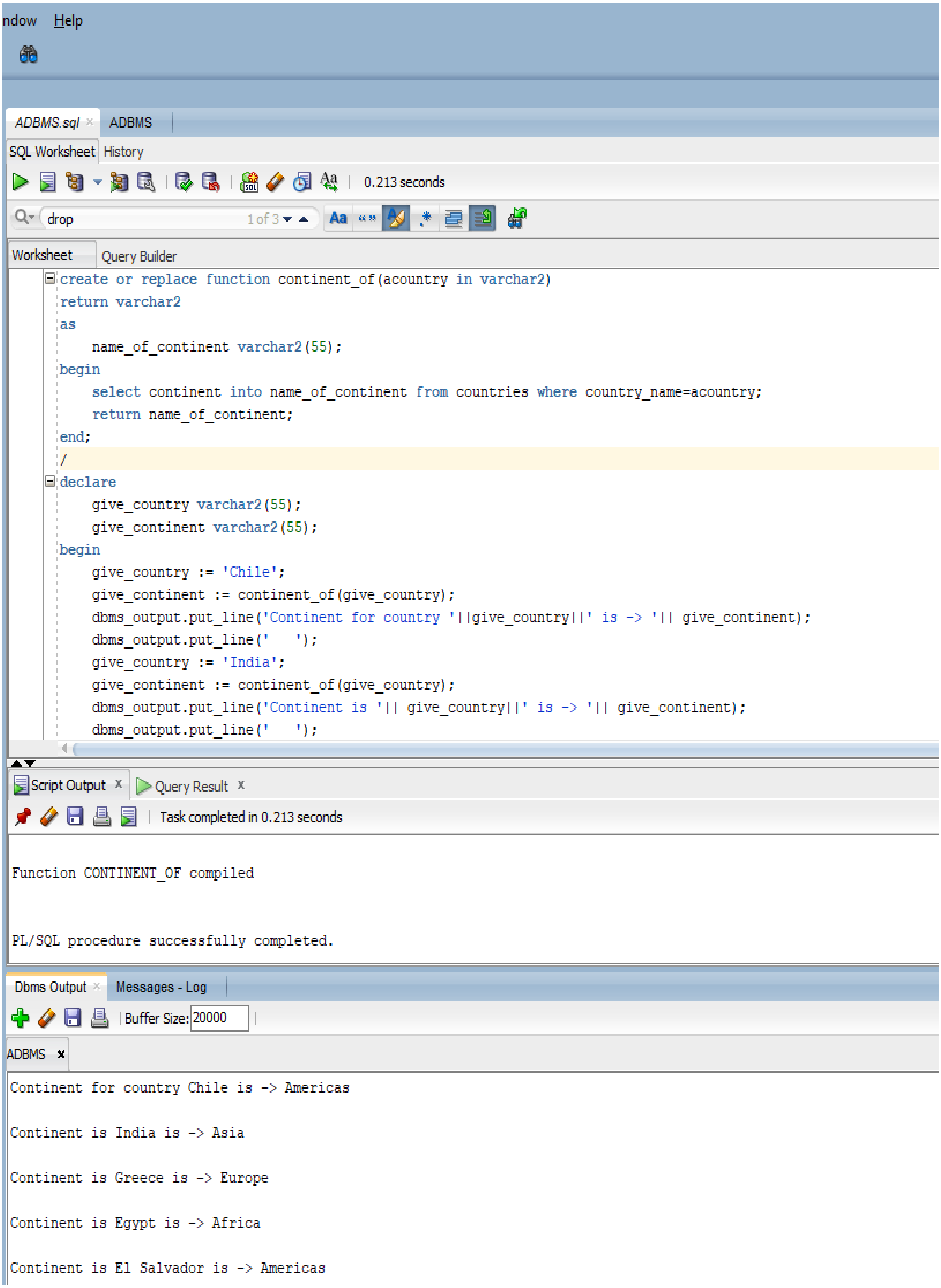
```
Continent for country Chile is -> Americas

Continent is India is -> Asia

Continent is Greece is -> Europe

Continent is Egypt is -> Africa

Continent is El Salvador is -> Americas
```



```
create or replace function continent_of(acountry in varchar2)
return varchar2
as
    name_of_continent varchar2(55);
begin
    select continent into name_of_continent from countries where country_name=acountry;
    return name_of_continent;
end;
/
declare
    give_country varchar2(55);
    give_continent varchar2(55);
begin
    give_country := 'Chile';
    give_continent := continent_of(give_country);
    dbms_output.put_line('Continent for country '||give_country||' is -> '|| give_continent);
    dbms_output.put_line('   ');
    give_country := 'India';
    give_continent := continent_of(give_country);
    dbms_output.put_line('Continent is '|| give_country||' is -> '|| give_continent);
    dbms_output.put_line('   ');
```

Task completed in 0.213 seconds

```
Function CONTINENT_OF compiled


PL/SQL procedure successfully completed.
```

Buffer Size: 20000

```
Continent for country Chile is -> Americas

Continent is India is -> Asia

Continent is Greece is -> Europe

Continent is Egypt is -> Africa

Continent is El Salvador is -> Americas
```

9) Write a PL/SQL program that calls the function CONTINENT-OF for every country in the table ADDITIONAL and inserts both the country and the continent it is in into the table CONT_COUNTRY. RECORD THE TRIGGER ACTIVATION MESSAGES AND SHOW THEM.

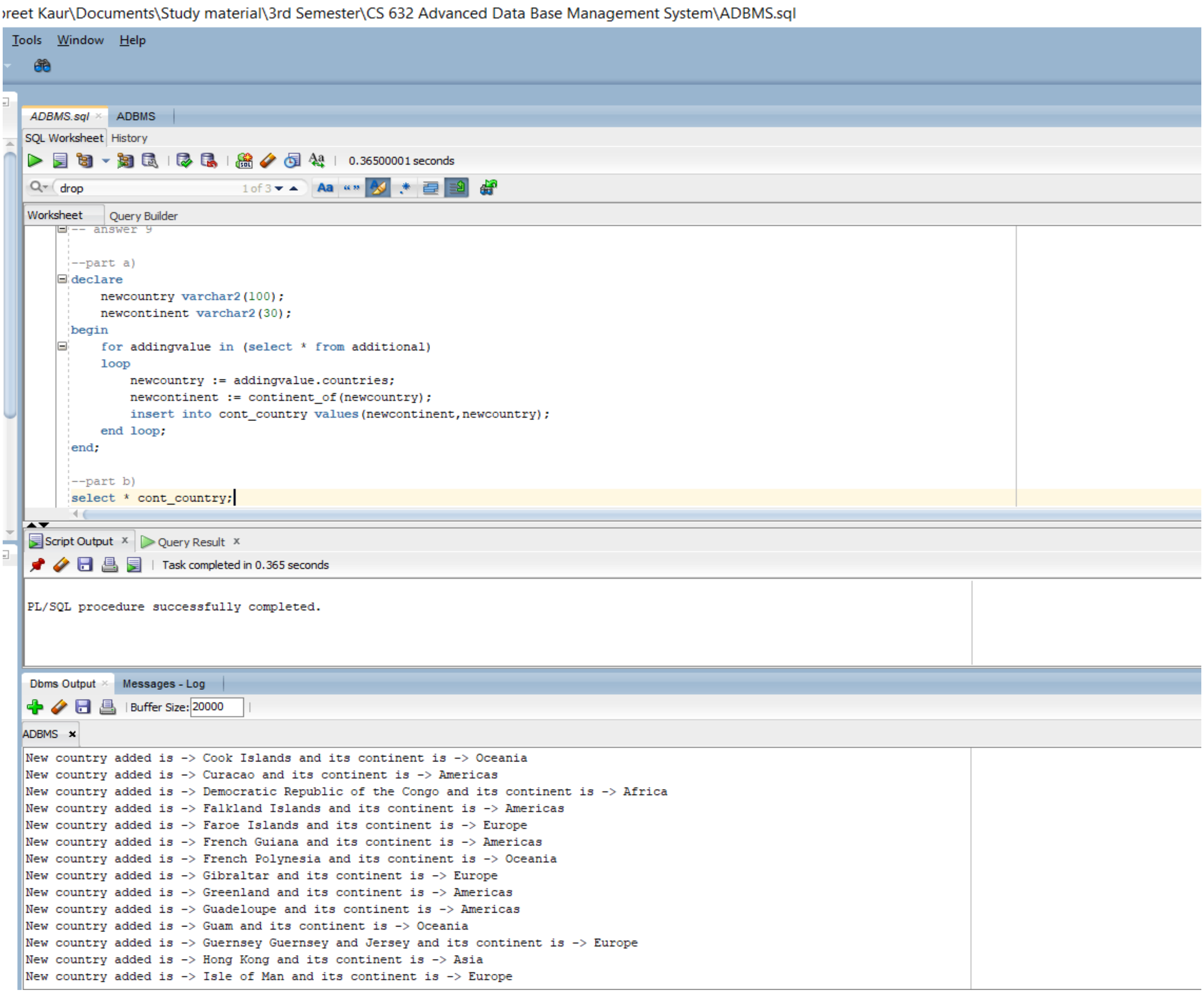
SHOW the program, show the final table CONT-COUNTRY.

Answer 9)

```
--part a)

declare
   newcountry varchar2(100);
   newcontinent varchar2(30);
begin
   for addingvalue in (select * from additional)
   loop
      newcountry := addingvalue.countries;
      newcontinent := continent_of(newcountry);
      insert into cont_country values(newcontinent,newcountry);
   end loop;
end;

--part b)
select * cont_country;
```



ɔreet Kaur\Documents\Study material\3rd Semester\CS 632 Advanced Data Base Management System\ADBMS.sql
Tools
Window
Help
ADBMS.sql
ADBMS
SQL Worksheet
History
0.36500001 seconds
drop
1 of 3
Worksheet
Query Builder

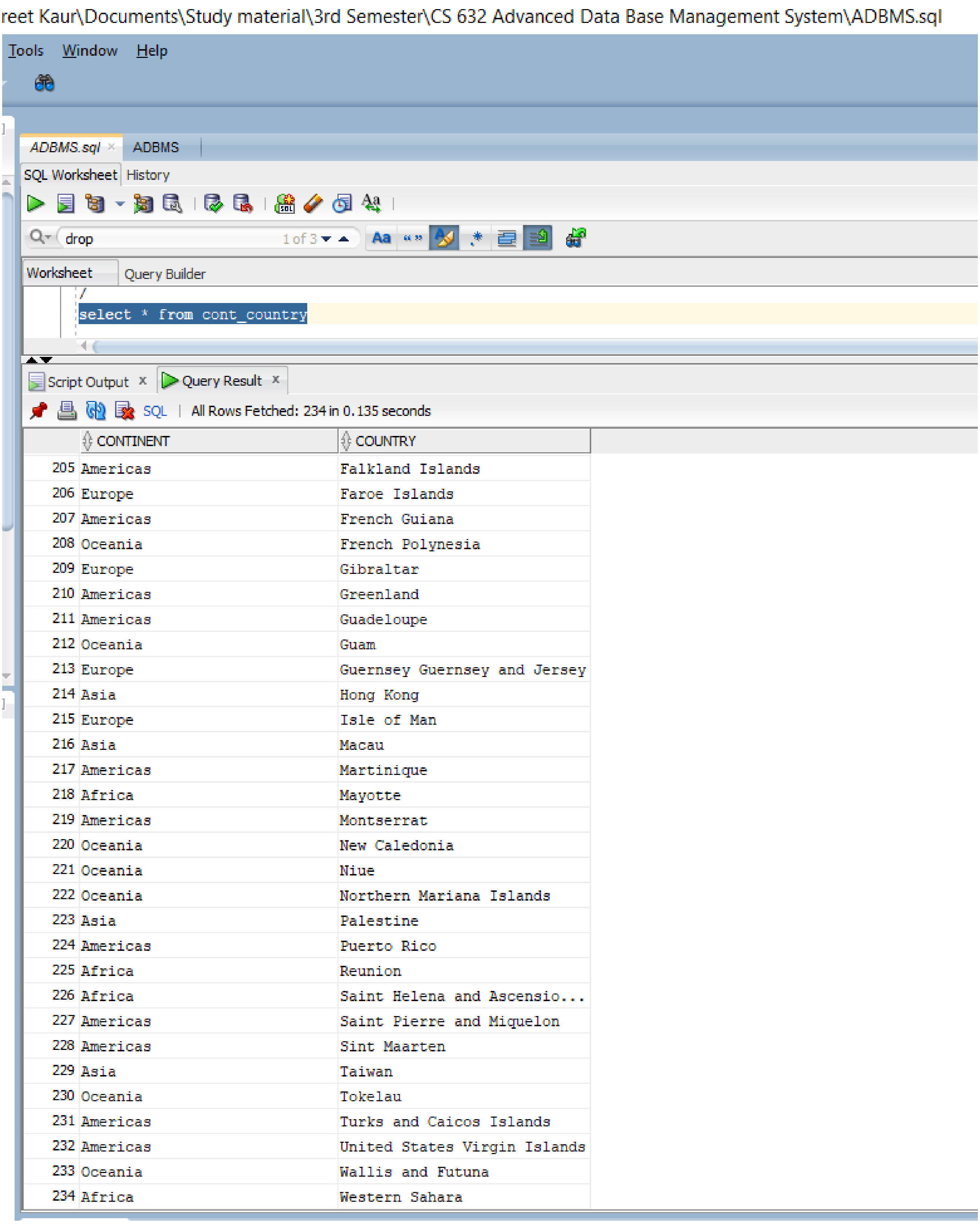
```
-- answer 9

--part a)
declare
    newcountry varchar2(100);
    newcontinent varchar2(30);
begin
    for addingvalue in (select * from additional)
    loop
        newcountry := addingvalue.countries;
        newcontinent := continent_of(newcountry);
        insert into cont_country values(newcontinent,newcountry);
    end loop;
end;

--part b)
select * cont_country;
```

```
PL/SQL procedure successfully completed.
```

```
New country added is -> Cook Islands and its continent is -> Oceania
New country added is -> Curacao and its continent is -> Americas
New country added is -> Democratic Republic of the Congo and its continent is -> Africa
New country added is -> Falkland Islands and its continent is -> Americas
New country added is -> Faroe Islands and its continent is -> Europe
New country added is -> French Guiana and its continent is -> Americas
New country added is -> French Polynesia and its continent is -> Oceania
New country added is -> Gibraltar and its continent is -> Europe
New country added is -> Greenland and its continent is -> Americas
New country added is -> Guadeloupe and its continent is -> Americas
New country added is -> Guam and its continent is -> Oceania
New country added is -> Guernsey Guernsey and Jersey and its continent is -> Europe
New country added is -> Hong Kong and its continent is -> Asia
New country added is -> Isle of Man and its continent is -> Europe
```



reet Kaur\Documents\Study material\3rd Semester\CS 632 Advanced Data Base Management System\ADBMS.sql

```
/
select * from cont_country
```

All Rows Fetched: 234 in 0.135 seconds

	CONTINENT	COUNTRY
205	Americas	Falkland Islands
206	Europe	Faroe Islands
207	Americas	French Guiana
208	Oceania	French Polynesia
209	Europe	Gibraltar
210	Americas	Greenland
211	Americas	Guadeloupe
212	Oceania	Guam
213	Europe	Guernsey Guernsey and Jersey
214	Asia	Hong Kong
215	Europe	Isle of Man
216	Asia	Macau
217	Americas	Martinique
218	Africa	Mayotte
219	Americas	Montserrat
220	Oceania	New Caledonia
221	Oceania	Niue
222	Oceania	Northern Mariana Islands
223	Asia	Palestine
224	Americas	Puerto Rico
225	Africa	Reunion
226	Africa	Saint Helena and Ascensio...
227	Americas	Saint Pierre and Miquelon
228	Americas	Sint Maarten
229	Asia	Taiwan
230	Oceania	Tokelau
231	Americas	Turks and Caicos Islands
232	Americas	United States Virgin Islands
233	Oceania	Wallis and Futuna
234	Africa	Western Sahara

THANKYOU