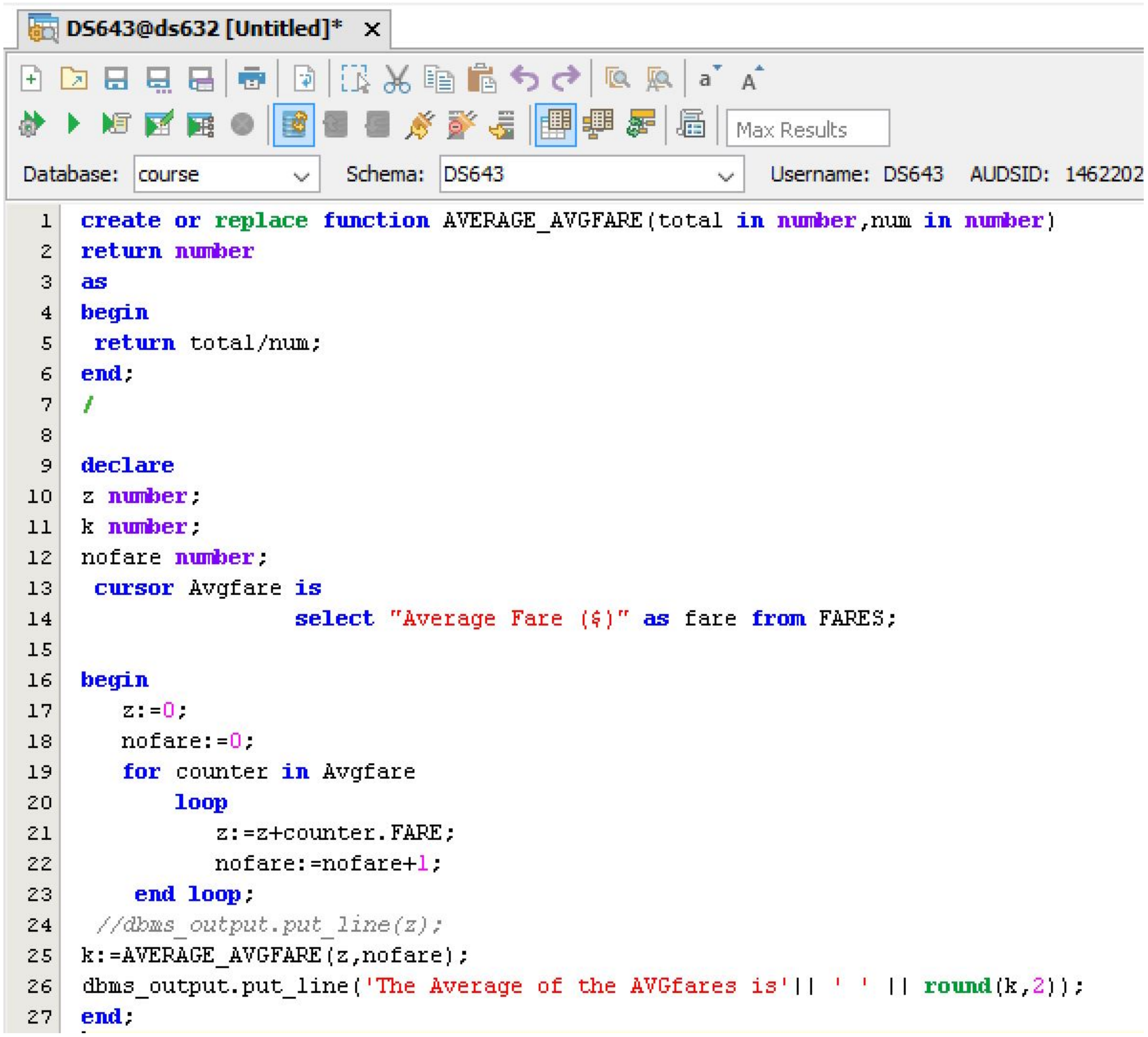
Homework 2
For
CS632-001
By
Dushyant Sankhla
Ds643@njit.edu

1. A) Write a PL/SQL program for the table FARES that computes the average (mean value) of all the Average Fairs



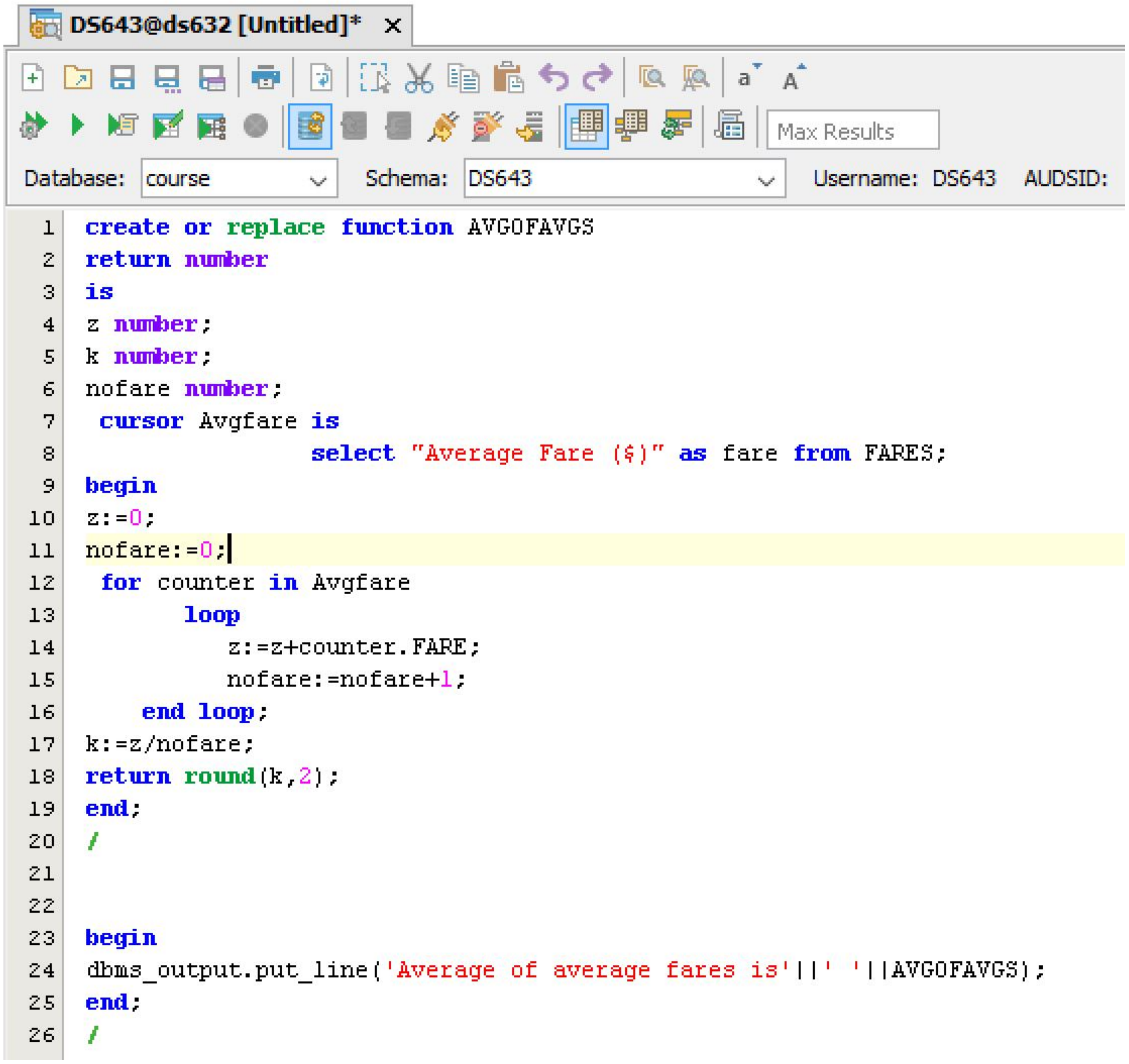
DS643@ds632 [Untitled]*

Database: course   Schema: DS643   Username: DS643   AUDSID: 1462202

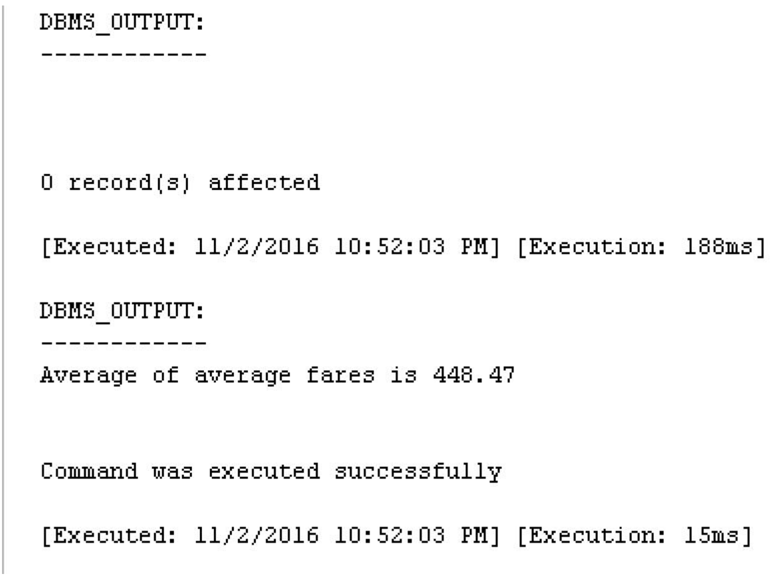
```
create or replace function AVERAGE_AVGFARE(total in number,num in number)
return number
as
begin
 return total/num;
end;
/

declare
z number;
k number;
nofare number;
 cursor Avgfare is
                select "Average Fare ($)" as fare from FARES;

begin
   z:=0;
   nofare:=0;
   for counter in Avgfare
       loop
           z:=z+counter.FARE;
           nofare:=nofare+1;
    end loop;
  //dbms_output.put_line(z);
k:=AVERAGE_AVGFARE(z,nofare);
dbms_output.put_line('The Average of the AVGfares is'|| ' ' || round(k,2));
end;
```

```
DBMS_OUTPUT:
------------



0 record(s) affected

[Executed: 11/2/2016 10:49:48 PM] [Execution: 16ms]

DBMS_OUTPUT:
------------
The Average of the AVGfares is 448.47


Command was executed successfully

[Executed: 11/2/2016 10:49:48 PM] [Execution: 234ms]
```

B) Transform a) into a function AVGOFAVGS with no arguments that returns as result the average of all Average Fairs.

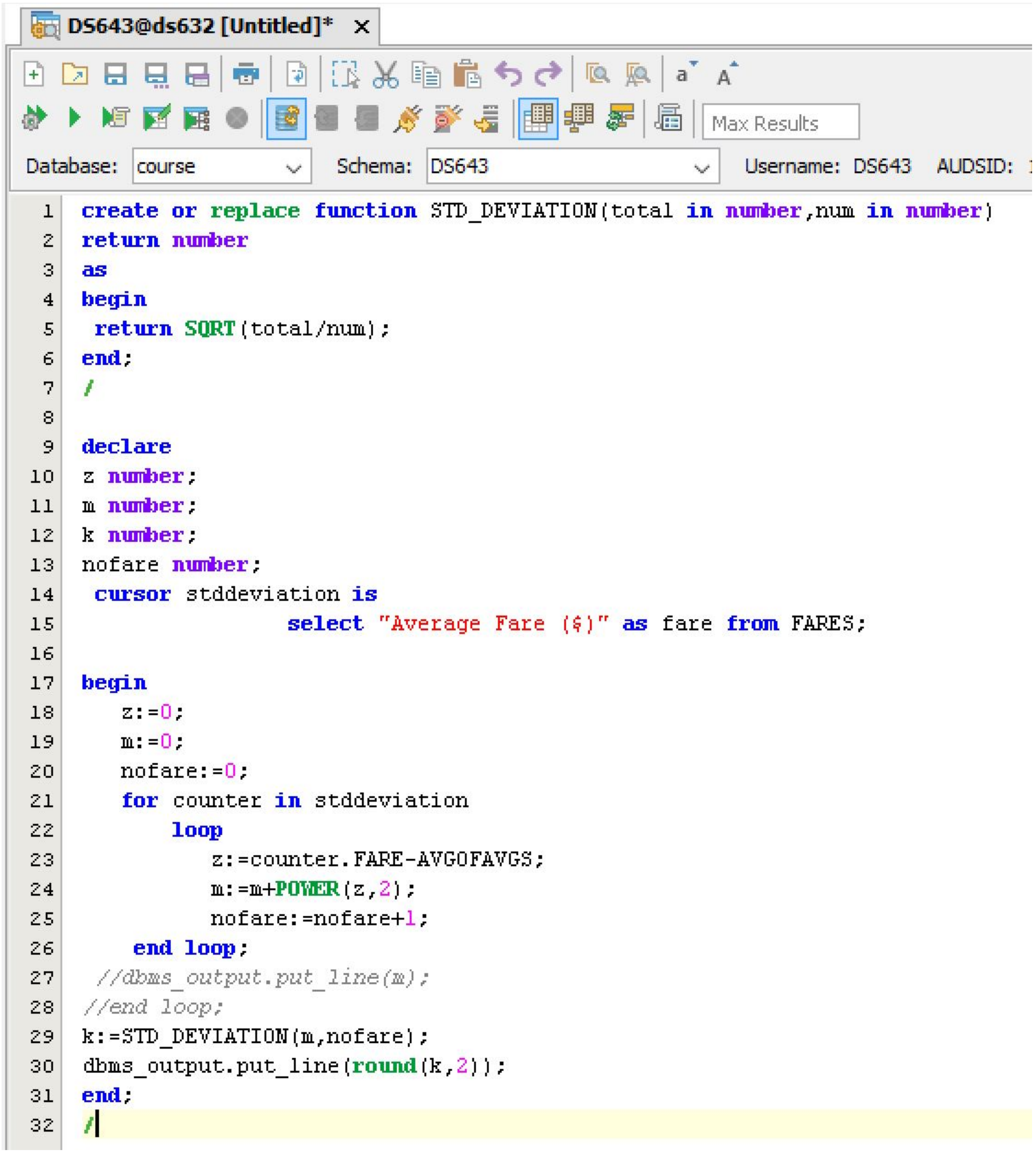


```
create or replace function AVGOFAVGS
return number
is
z number;
k number;
nofare number;
 cursor Avgfare is
                select "Average Fare ($)" as fare from FARES;
begin
z:=0;
nofare:=0;
 for counter in Avgfare
       loop
          z:=z+counter.FARE;
          nofare:=nofare+1;
    end loop;
k:=z/nofare;
return round(k,2);
end;
/


begin
dbms_output.put_line('Average of average fares is'||' '||AVGOFAVGS);
end;
/
```

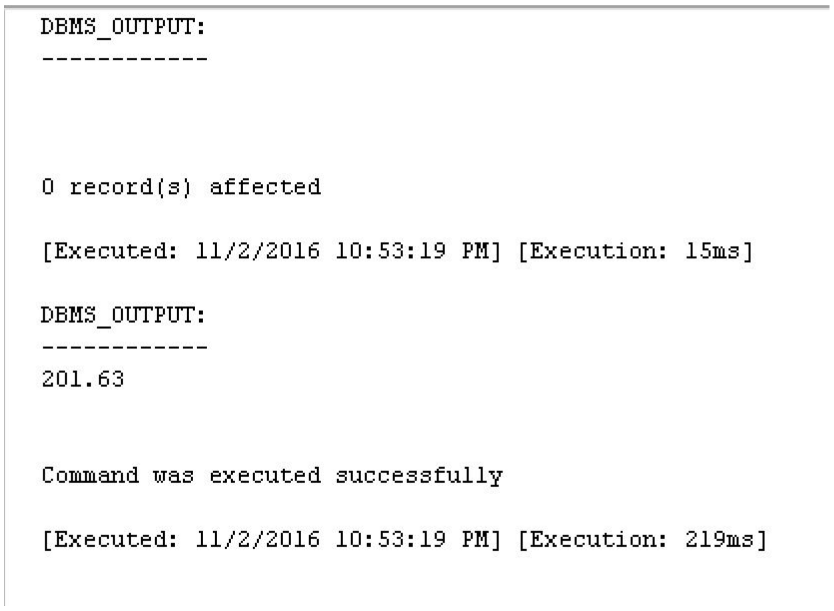


```
DBMS_OUTPUT:
------------



0 record(s) affected

[Executed: 11/2/2016 10:52:03 PM] [Execution: 188ms]

DBMS_OUTPUT:
------------
Average of average fares is 448.47


Command was executed successfully

[Executed: 11/2/2016 10:52:03 PM] [Execution: 15ms]
```

C) Write a PL/SQL program that computes the Standard Deviation of the Average Fares

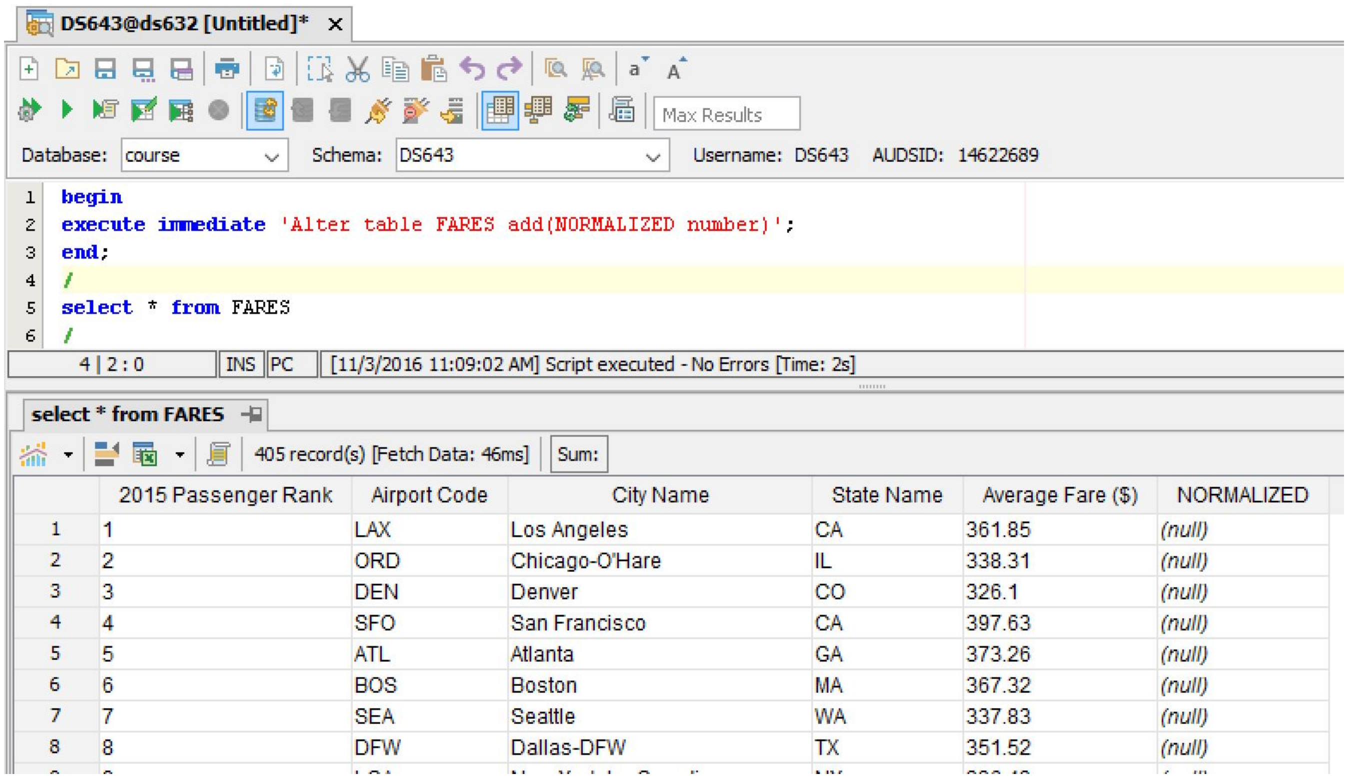


```
create or replace function STD_DEVIATION(total in number,num in number)
return number
as
begin
 return SQRT(total/num);
end;
/

declare
z number;
m number;
k number;
nofare number;
 cursor stddeviation is
                select "Average Fare ($)" as fare from FARES;

begin
   z:=0;
   m:=0;
   nofare:=0;
   for counter in stddeviation
       loop
          z:=counter.FARE-AVGOFAVGS;
          m:=m+POWER(z,2);
          nofare:=nofare+1;
    end loop;
 //dbms_output.put_line(m);
//end loop;
k:=STD_DEVIATION(m,nofare);
dbms_output.put_line(round(k,2));
end;
/
```



```
DBMS_OUTPUT:
------------



0 record(s) affected

[Executed: 11/2/2016 10:53:19 PM] [Execution: 15ms]

DBMS_OUTPUT:
------------
201.63


Command was executed successfully

[Executed: 11/2/2016 10:53:19 PM] [Execution: 219ms]
```

D) Write a **PL/SQL program** that executes only one ALTER command that will add a column of type number to the table FARES. The name of the new column should be NORMALIZED



DS643@ds632 [Untitled]*

Database: course   Schema: DS643   Username: DS643   AUDSID: 14622689

```
begin
execute immediate 'Alter table FARES add(NORMALIZED number)';
end;
/
select * from FARES
/
```

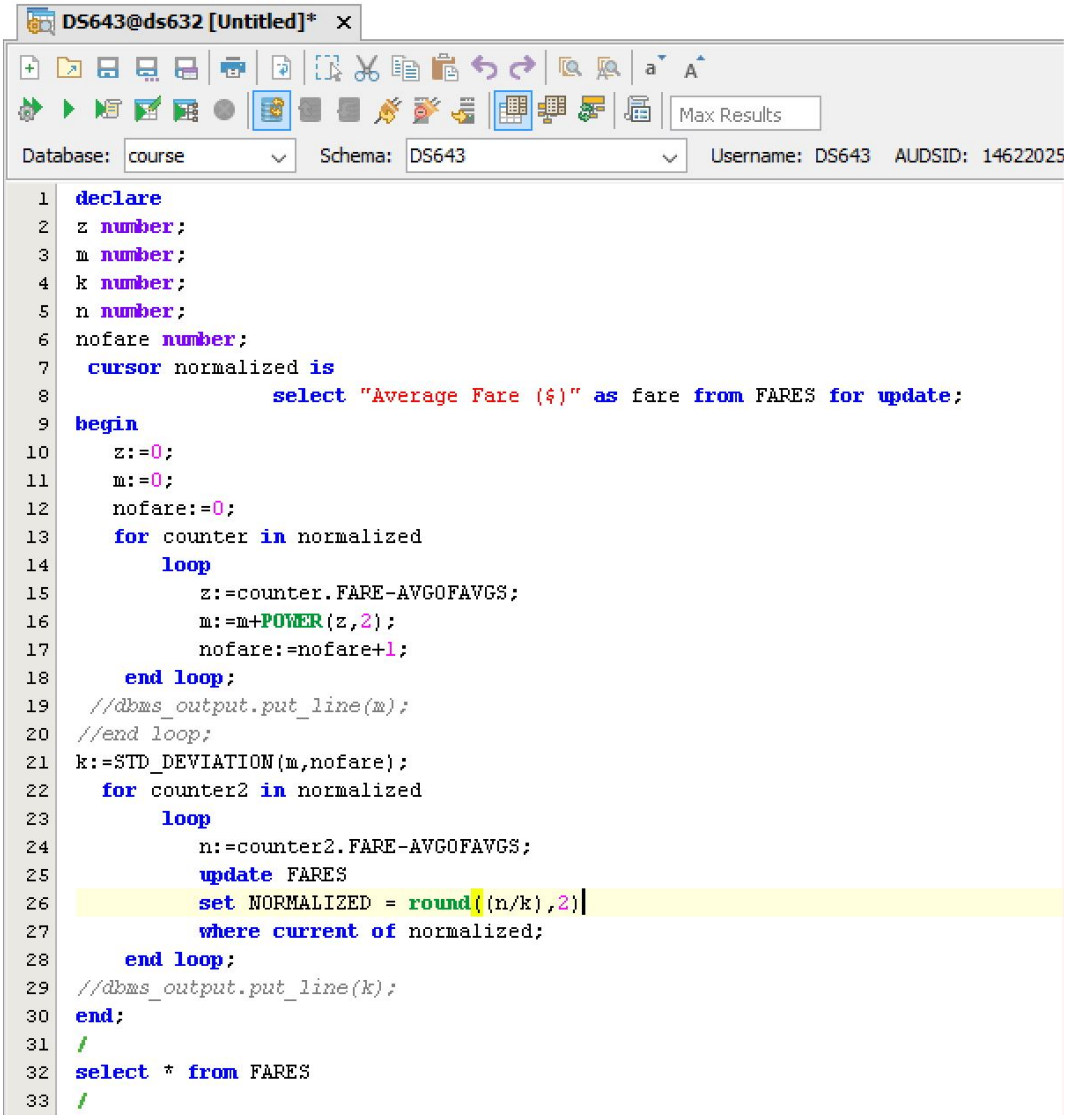
```
4 | 2 : 0   INS  PC   [11/3/2016 11:09:02 AM] Script executed - No Errors [Time: 2s]
```

select * from FARES

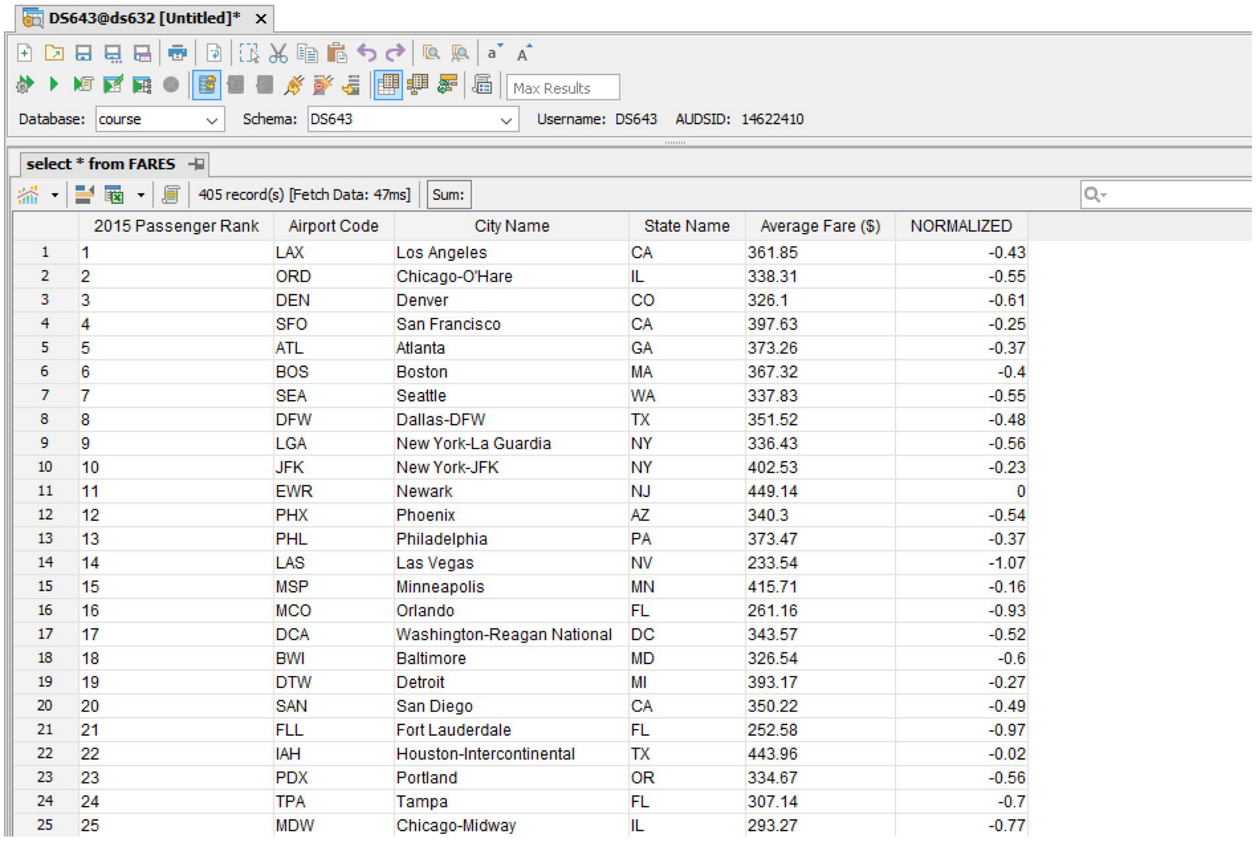
405 record(s) [Fetch Data: 46ms]   Sum:

	2015 Passenger Rank	Airport Code	City Name	State Name	Average Fare ($)	NORMALIZED
1	1	LAX	Los Angeles	CA	361.85	(null)
2	2	ORD	Chicago-O'Hare	IL	338.31	(null)
3	3	DEN	Denver	CO	326.1	(null)
4	4	SFO	San Francisco	CA	397.63	(null)
5	5	ATL	Atlanta	GA	373.26	(null)
6	6	BOS	Boston	MA	367.32	(null)
7	7	SEA	Seattle	WA	337.83	(null)
8	8	DFW	Dallas-DFW	TX	351.52	(null)

E) Write a PL/SQL program that puts the Normalized value of each AVG into the column NORMALIZED.



```
declare
z number;
m number;
k number;
n number;
nofare number;
 cursor normalized is
               select "Average Fare ($)" as fare from FARES for update;
begin
   z:=0;
   m:=0;
   nofare:=0;
   for counter in normalized
       loop
          z:=counter.FARE-AVGOFAVGS;
          m:=m+POWER(z,2);
          nofare:=nofare+1;
     end loop;
 //dbms_output.put_line(m);
//end loop;
k:=STD_DEVIATION(m,nofare);
  for counter2 in normalized
       loop
          n:=counter2.FARE-AVGOFAVGS;
          update FARES
          set NORMALIZED = round((n/k),2)
          where current of normalized;
     end loop;
//dbms_output.put_line(k);
end;
/
select * from FARES
/
```



DS643@ds632 [Untitled]*

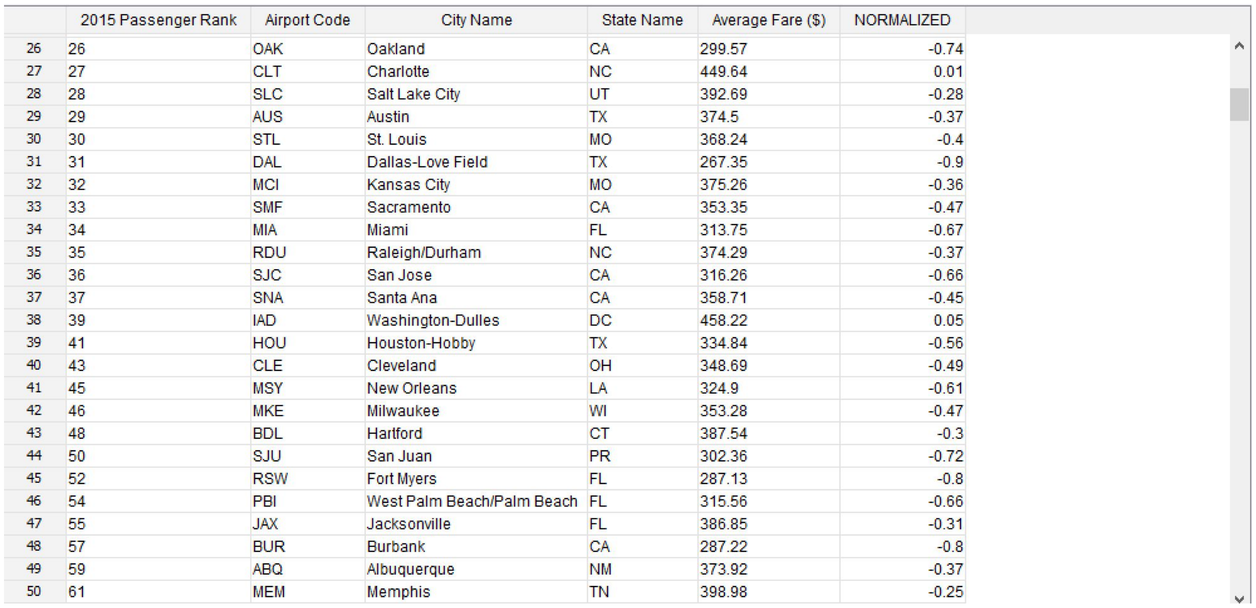
Max Results

Database: course Schema: DS643 Username: DS643 AUDSID: 14622410

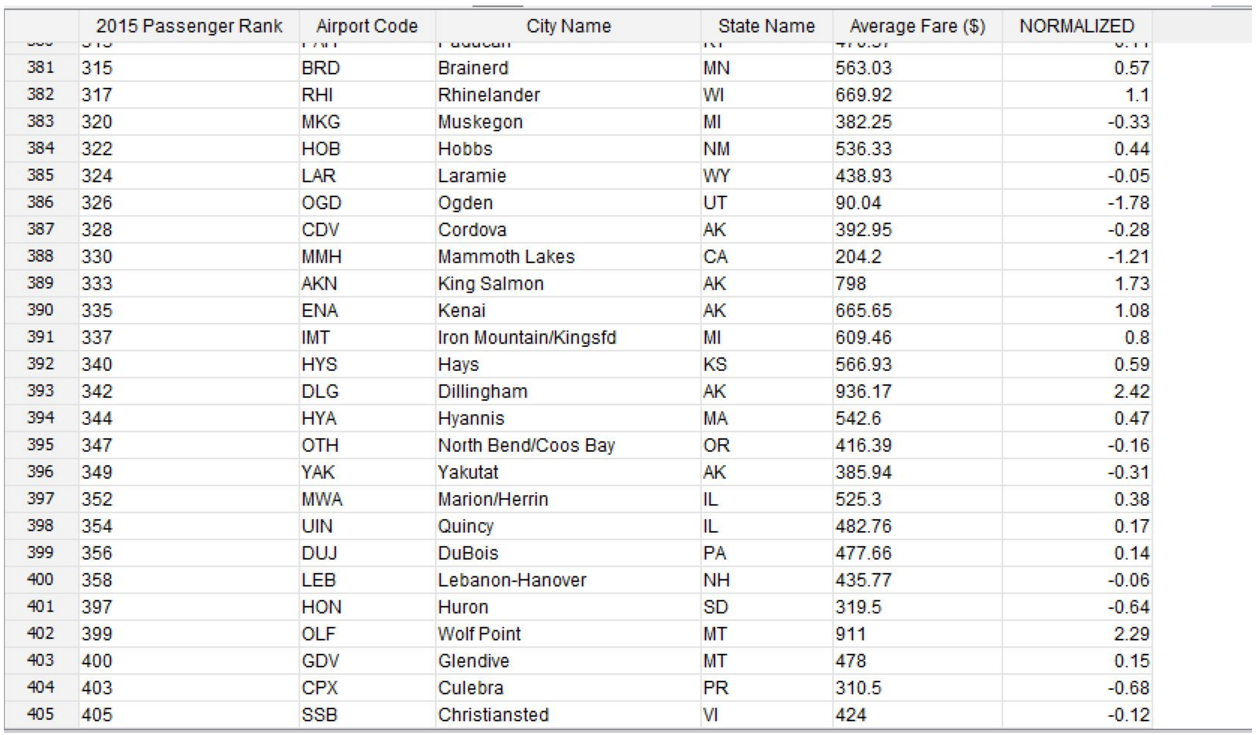
select * from FARES

405 record(s) [Fetch Data: 47ms] Sum:

	2015 Passenger Rank	Airport Code	City Name	State Name	Average Fare ($)	NORMALIZED
1	1	LAX	Los Angeles	CA	361.85	-0.43
2	2	ORD	Chicago-O'Hare	IL	338.31	-0.55
3	3	DEN	Denver	CO	326.1	-0.61
4	4	SFO	San Francisco	CA	397.63	-0.25
5	5	ATL	Atlanta	GA	373.26	-0.37
6	6	BOS	Boston	MA	367.32	-0.4
7	7	SEA	Seattle	WA	337.83	-0.55
8	8	DFW	Dallas-DFW	TX	351.52	-0.48
9	9	LGA	New York-La Guardia	NY	336.43	-0.56
10	10	JFK	New York-JFK	NY	402.53	-0.23
11	11	EWR	Newark	NJ	449.14	0
12	12	PHX	Phoenix	AZ	340.3	-0.54
13	13	PHL	Philadelphia	PA	373.47	-0.37
14	14	LAS	Las Vegas	NV	233.54	-1.07
15	15	MSP	Minneapolis	MN	415.71	-0.16
16	16	MCO	Orlando	FL	261.16	-0.93
17	17	DCA	Washington-Reagan National	DC	343.57	-0.52
18	18	BWI	Baltimore	MD	326.54	-0.6
19	19	DTW	Detroit	MI	393.17	-0.27
20	20	SAN	San Diego	CA	350.22	-0.49
21	21	FLL	Fort Lauderdale	FL	252.58	-0.97
22	22	IAH	Houston-Intercontinental	TX	443.96	-0.02
23	23	PDX	Portland	OR	334.67	-0.56
24	24	TPA	Tampa	FL	307.14	-0.7
25	25	MDW	Chicago-Midway	IL	293.27	-0.77

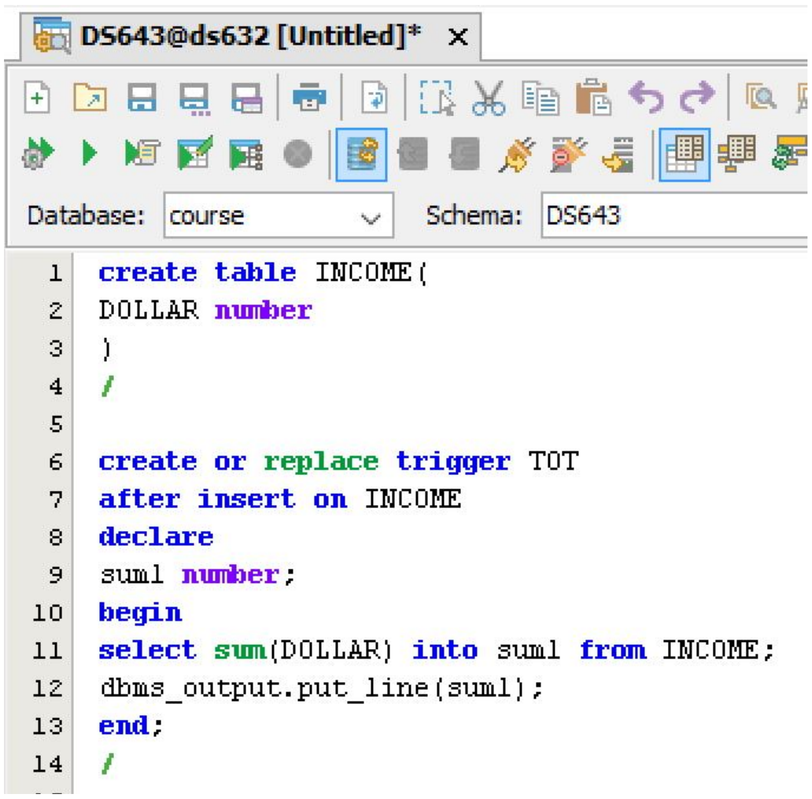


	2015 Passenger Rank	Airport Code	City Name	State Name	Average Fare ($)	NORMALIZED
26	26	OAK	Oakland	CA	299.57	-0.74
27	27	CLT	Charlotte	NC	449.64	0.01
28	28	SLC	Salt Lake City	UT	392.69	-0.28
29	29	AUS	Austin	TX	374.5	-0.37
30	30	STL	St. Louis	MO	368.24	-0.4
31	31	DAL	Dallas-Love Field	TX	267.35	-0.9
32	32	MCI	Kansas City	MO	375.26	-0.36
33	33	SMF	Sacramento	CA	353.35	-0.47
34	34	MIA	Miami	FL	313.75	-0.67
35	35	RDU	Raleigh/Durham	NC	374.29	-0.37
36	36	SJC	San Jose	CA	316.26	-0.66
37	37	SNA	Santa Ana	CA	358.71	-0.45
38	39	IAD	Washington-Dulles	DC	458.22	0.05
39	41	HOU	Houston-Hobby	TX	334.84	-0.56
40	43	CLE	Cleveland	OH	348.69	-0.49
41	45	MSY	New Orleans	LA	324.9	-0.61
42	46	MKE	Milwaukee	WI	353.28	-0.47
43	48	BDL	Hartford	CT	387.54	-0.3
44	50	SJU	San Juan	PR	302.36	-0.72
45	52	RSW	Fort Myers	FL	287.13	-0.8
46	54	PBI	West Palm Beach/Palm Beach	FL	315.56	-0.66
47	55	JAX	Jacksonville	FL	386.85	-0.31
48	57	BUR	Burbank	CA	287.22	-0.8
49	59	ABQ	Albuquerque	NM	373.92	-0.37
50	61	MEM	Memphis	TN	398.98	-0.25



	2015 Passenger Rank	Airport Code	City Name	State Name	Average Fare ($)	NORMALIZED
381	315	BRD	Brainerd	MN	563.03	0.57
382	317	RHI	Rhinelander	WI	669.92	1.1
383	320	MKG	Muskegon	MI	382.25	-0.33
384	322	HOB	Hobbs	NM	536.33	0.44
385	324	LAR	Laramie	WY	438.93	-0.05
386	326	OGD	Ogden	UT	90.04	-1.78
387	328	CDV	Cordova	AK	392.95	-0.28
388	330	MMH	Mammoth Lakes	CA	204.2	-1.21
389	333	AKN	King Salmon	AK	798	1.73
390	335	ENA	Kenai	AK	665.65	1.08
391	337	IMT	Iron Mountain/Kingsfd	MI	609.46	0.8
392	340	HYS	Hays	KS	566.93	0.59
393	342	DLG	Dillingham	AK	936.17	2.42
394	344	HYA	Hyannis	MA	542.6	0.47
395	347	OTH	North Bend/Coos Bay	OR	416.39	-0.16
396	349	YAK	Yakutat	AK	385.94	-0.31
397	352	MWA	Marion/Herrin	IL	525.3	0.38
398	354	UIN	Quincy	IL	482.76	0.17
399	356	DUJ	DuBois	PA	477.66	0.14
400	358	LEB	Lebanon-Hanover	NH	435.77	-0.06
401	397	HON	Huron	SD	319.5	-0.64
402	399	OLF	Wolf Point	MT	911	2.29
403	400	GDV	Glendive	MT	478	0.15
404	403	CPX	Culebra	PR	310.5	-0.68
405	405	SSB	Christiansted	VI	424	-0.12

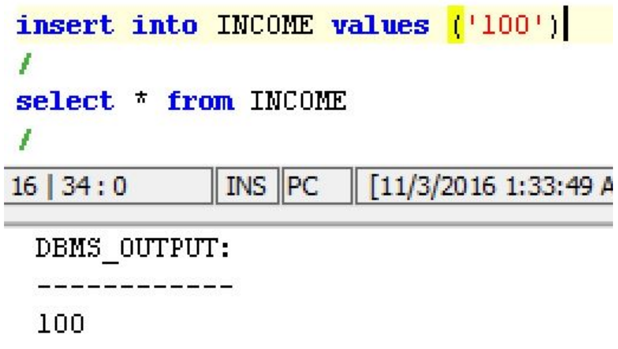
2. A) Create a new table INCOME with only one column called DOLLAR. Create an insert trigger TOT that will tell you the sum of all the values in the column DOLLAR.



DS643@ds632 [Untitled]*

Database: course   Schema: DS643

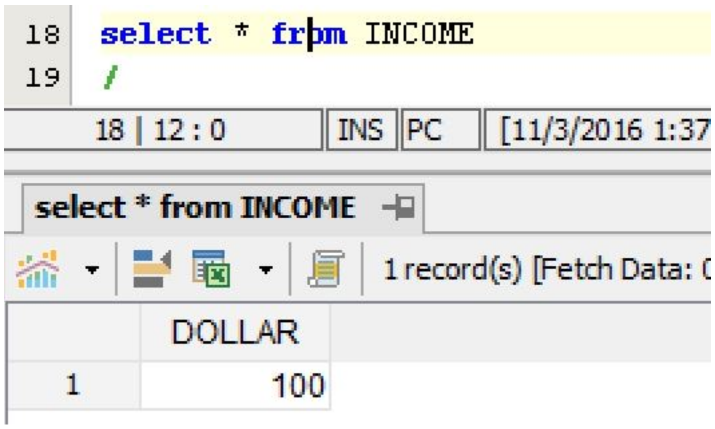
```sql
create table INCOME(
DOLLAR number
)
/

create or replace trigger TOT
after insert on INCOME
declare
sum1 number;
begin
select sum(DOLLAR) into sum1 from INCOME;
dbms_output.put_line(sum1);
end;
/
```



```
insert into INCOME values ('100')
/
select * from INCOME
/
```

16 | 34 : 0 | INS | PC | [11/3/2016 1:33:49 A

```
DBMS_OUTPUT:
------------
100
```



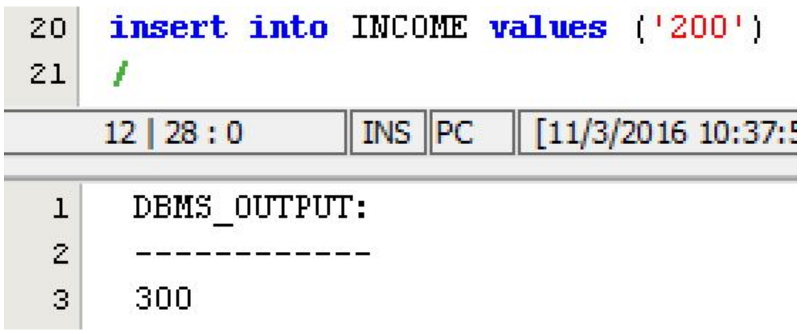
```
select * from INCOME
/
```

18 | 12 : 0 | INS | PC | [11/3/2016 1:37

select * from INCOME

1 record(s) [Fetch Data: (

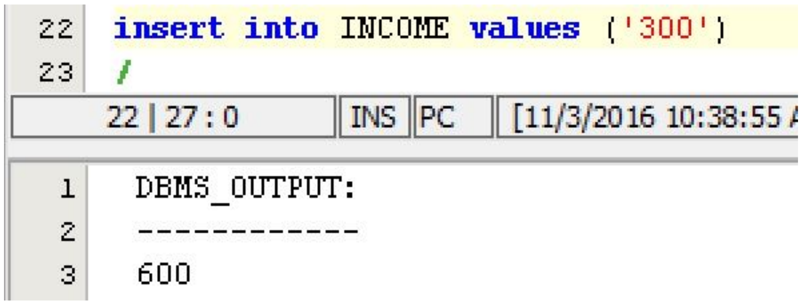
	DOLLAR
1	100



```
insert into INCOME values ('200')
/
```

12 | 28 : 0 | INS | PC | [11/3/2016 10:37:5

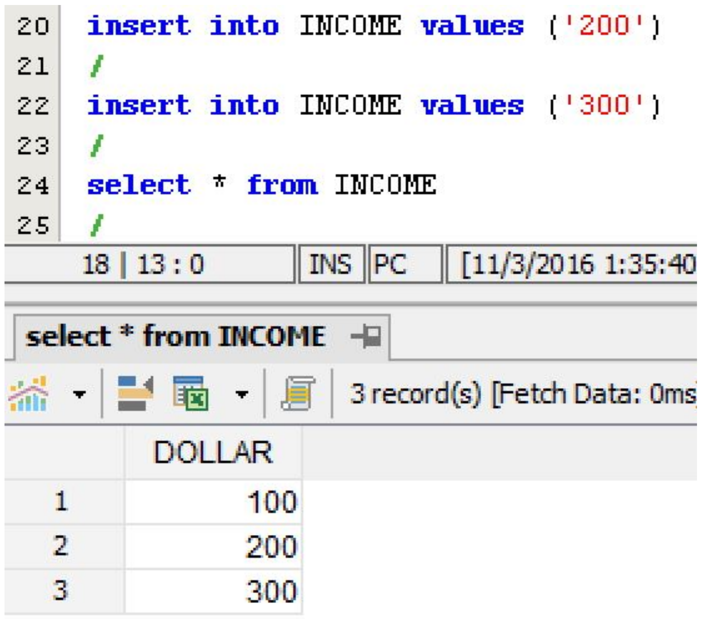
```
DBMS_OUTPUT:
------------
300
```



```
insert into INCOME values ('300')
/
```

22 | 27 : 0 | INS | PC | [11/3/2016 10:38:55 A

```
DBMS_OUTPUT:
------------
600
```



```
insert into INCOME values ('200')
/
insert into INCOME values ('300')
/
select * from INCOME
/
```

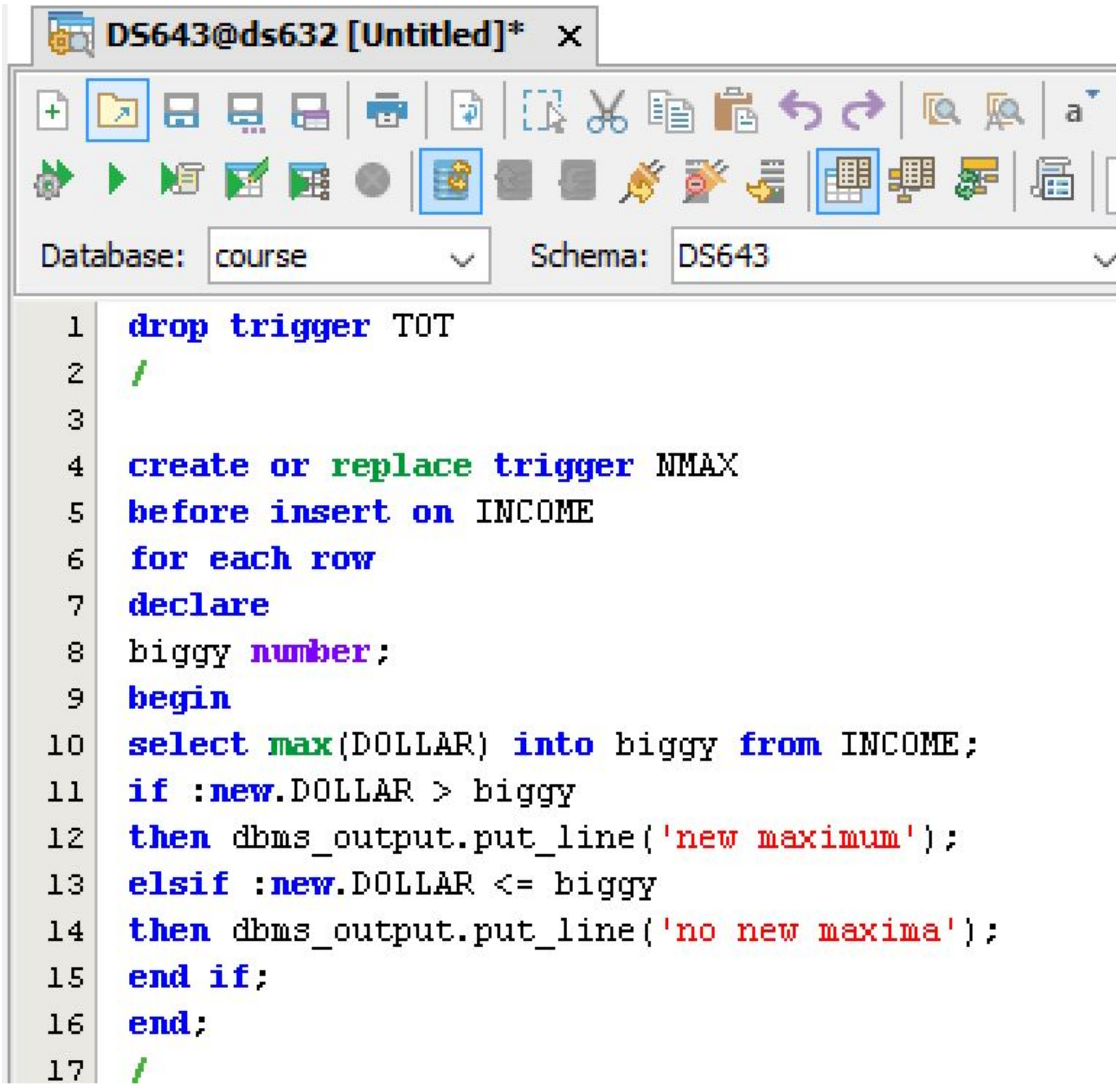
18 | 13 : 0 | INS | PC | [11/3/2016 1:35:40

select * from INCOME

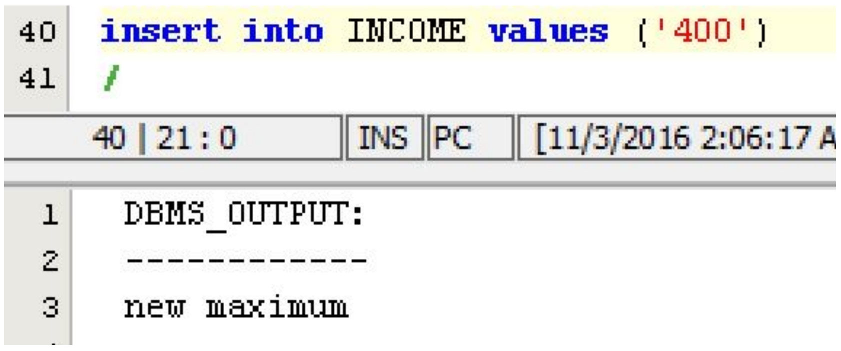
3 record(s) [Fetch Data: 0ms

	DOLLAR
1	100
2	200
3	300

B) Drop the trigger from 2)a .Create a new trigger NMAX on the table INCOME. If the number you are inserting is greater than the LARGEST number in the table before the insertion, the trigger should send the message 'new maximum' to the screen.If the number you are insertingis smaller or equal to the largest number in the table before the insertion, the trigger should send the message'no new maximum' to the screen.



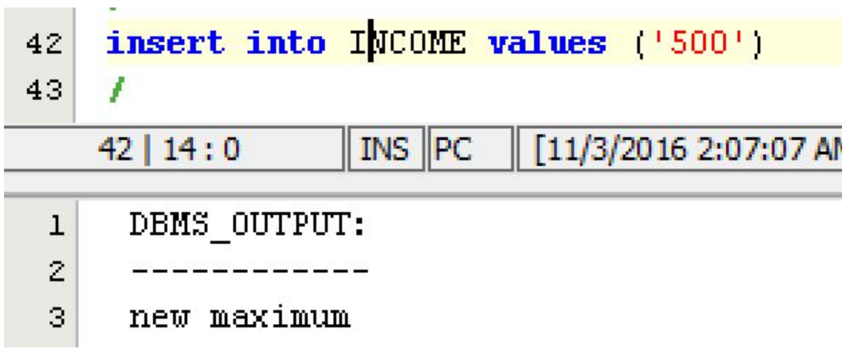
```
drop trigger TOT
/

create or replace trigger NMAX
before insert on INCOME
for each row
declare
biggy number;
begin
select max(DOLLAR) into biggy from INCOME;
if :new.DOLLAR > biggy
then dbms_output.put_line('new maximum');
elsif :new.DOLLAR <= biggy
then dbms_output.put_line('no new maxima');
end if;
end;
/
```



```
insert into INCOME values ('400')
/
```

```
40 | 21 : 0   INS   PC   [11/3/2016 2:06:17 A
```

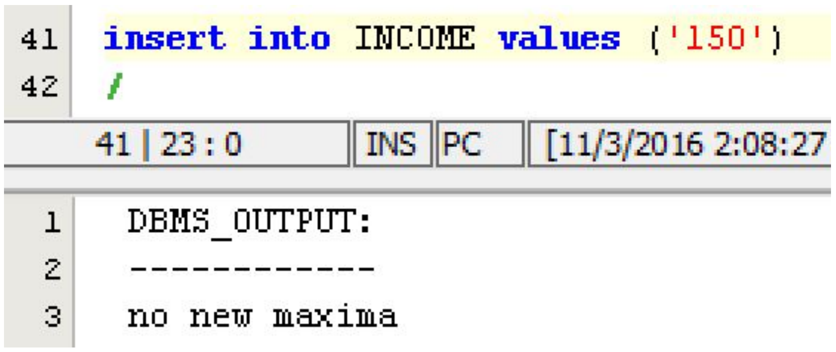
```
DBMS_OUTPUT:
------------
new maximum
```



```
insert into INCOME values ('500')
/
```

```
42 | 14 : 0   INS   PC   [11/3/2016 2:07:07 AI
```

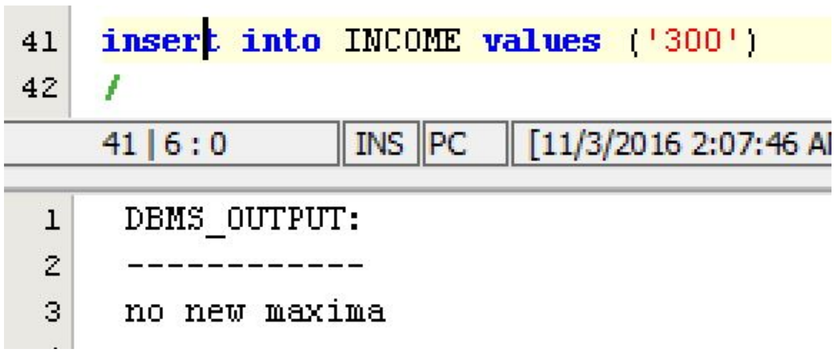
```
DBMS_OUTPUT:
------------
new maximum
```



```
insert into INCOME values ('150')
/
```

```
41 | 23 : 0   INS   PC   [11/3/2016 2:08:27
```

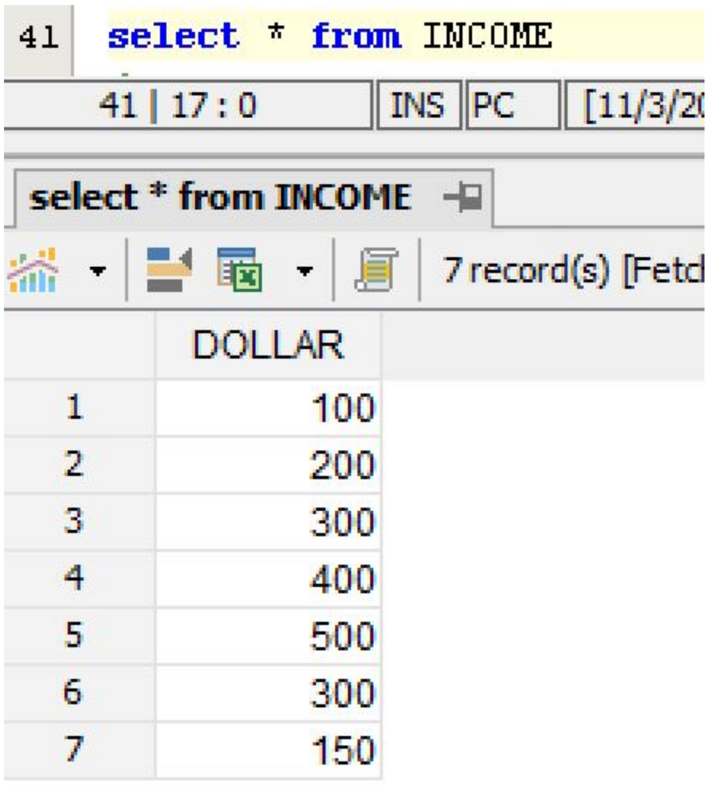
```
DBMS_OUTPUT:
------------
no new maxima
```



```
insert into INCOME values ('300')
/
```

```
41 | 6 : 0   INS   PC   [11/3/2016 2:07:46 AI
```

```
DBMS_OUTPUT:
------------
no new maxima
```



```
select * from INCOME
```

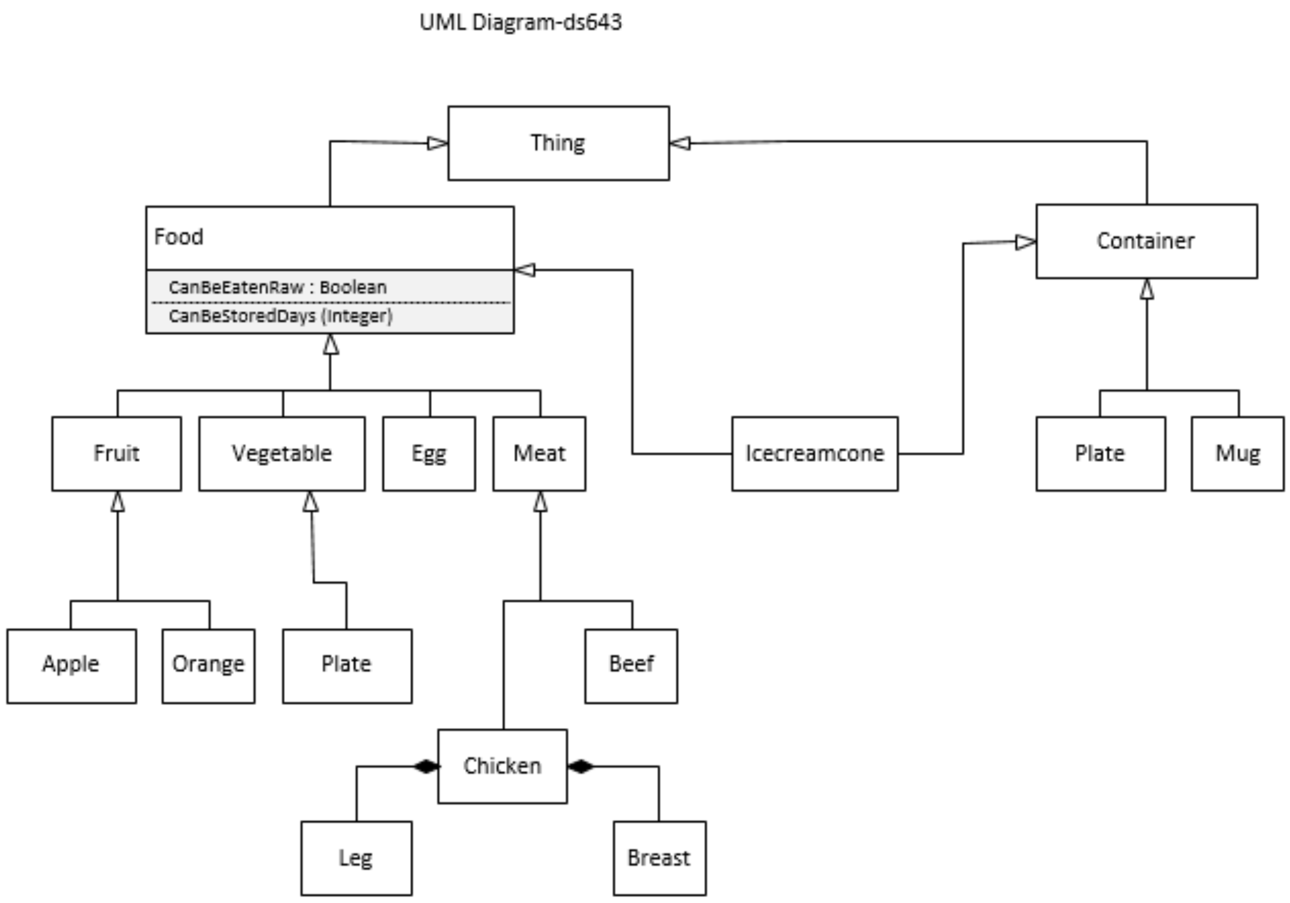
41 | 17 : 0 | INS | PC | [11/3/2

select * from INCOME

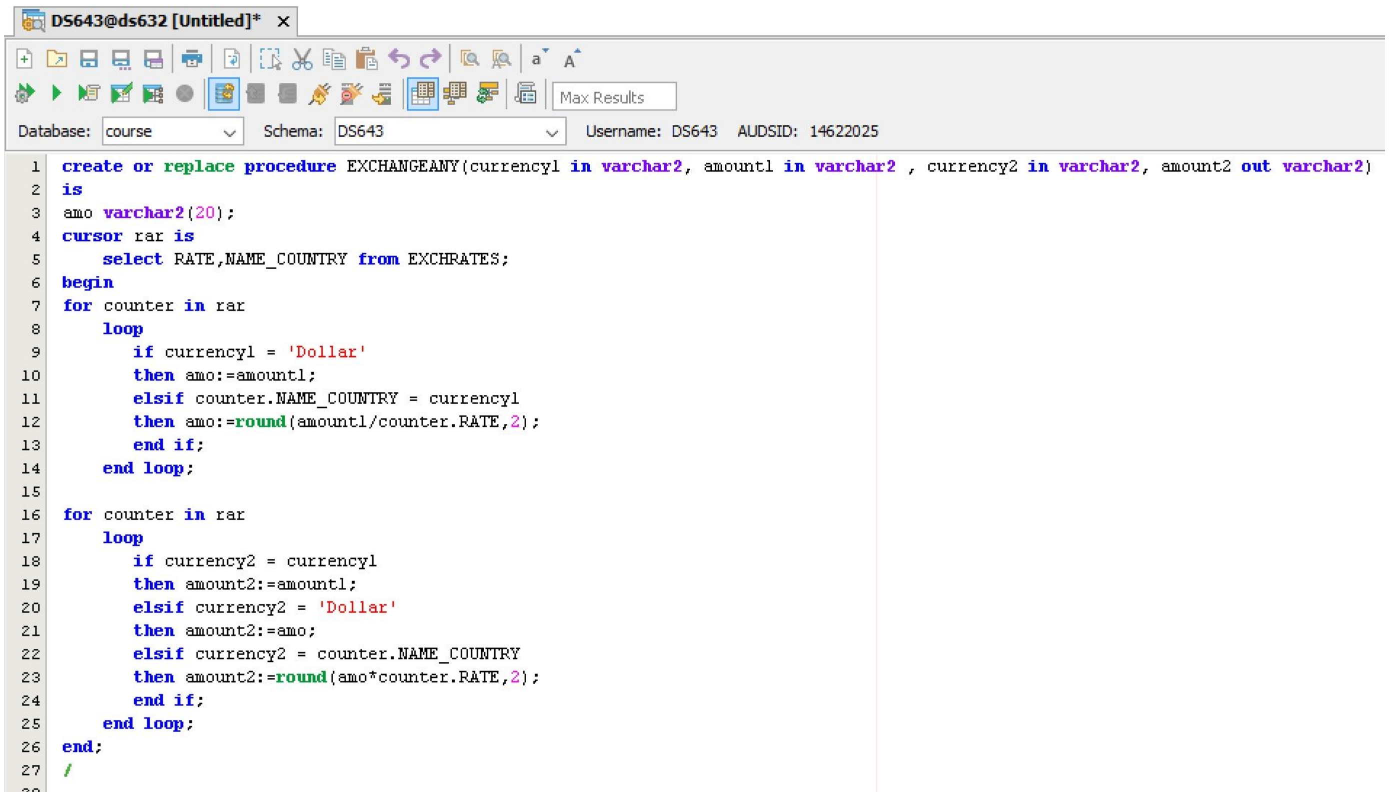
7 record(s) [Fetc

	DOLLAR
1	100
2	200
3	300
4	400
5	500
6	300
7	150

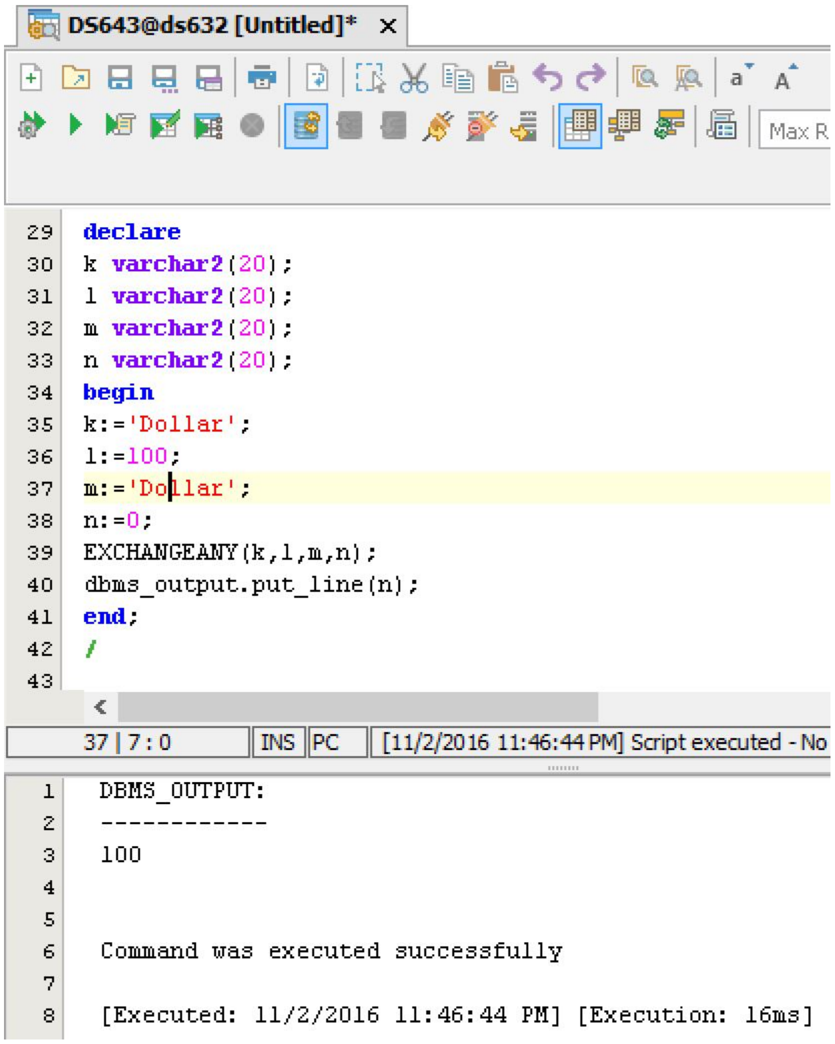
3. Draw a UML diagram expressing the following knowledge (using VISISO).



4. Using the table EXCHRATE write a procedure EXCHANGEANY that takes four arguments EXCHANGEANY(currency1, amount1, currency2, amount2).This function should translate currency1 into dollars using the EXCHRATES table. It should then translate the dollars into currency2, again using the EXCHRATES table.Note that amount2 should be an out parameter.



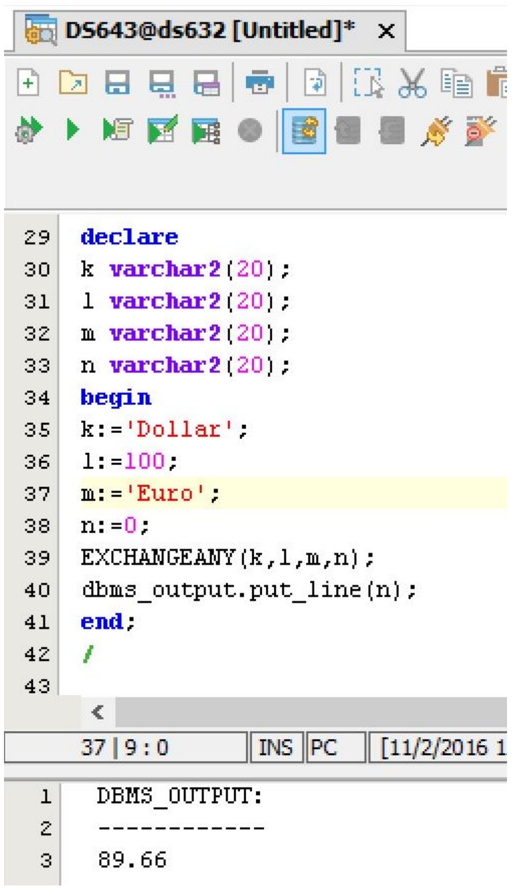
```
create or replace procedure EXCHANGEANY(currency1 in varchar2, amount1 in varchar2 , currency2 in varchar2, amount2 out varchar2)
is
amo varchar2(20);
cursor rar is
    select RATE,NAME_COUNTRY from EXCHRATES;
begin
for counter in rar
    loop
       if currency1 = 'Dollar'
       then amo:=amount1;
       elsif counter.NAME_COUNTRY = currency1
       then amo:=round(amount1/counter.RATE,2);
       end if;
    end loop;

for counter in rar
    loop
       if currency2 = currency1
       then amount2:=amount1;
       elsif currency2 = 'Dollar'
       then amount2:=amo;
       elsif currency2 = counter.NAME_COUNTRY
       then amount2:=round(amo*counter.RATE,2);
       end if;
    end loop;
end;
/
```



```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Dollar';
l:=100;
m:='Dollar';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

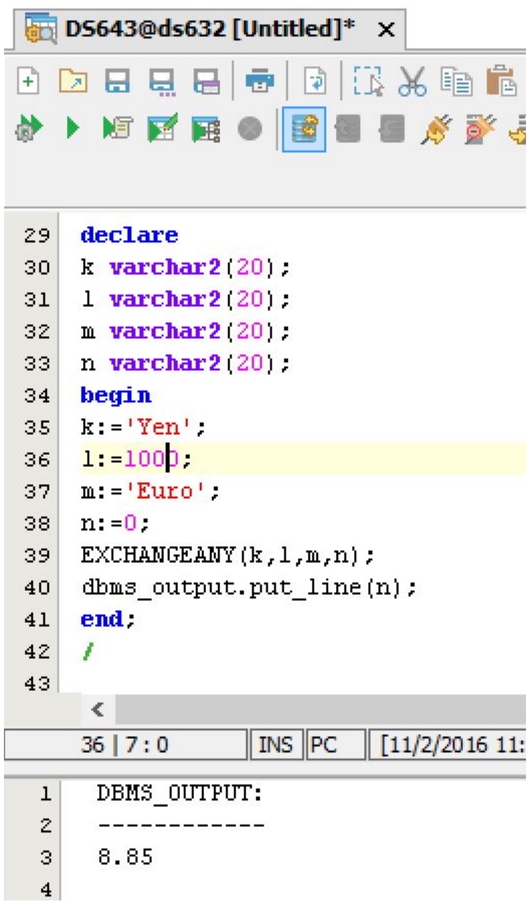
```
37 | 7 : 0    INS  PC   [11/2/2016 11:46:44 PM] Script executed - No
```

```
DBMS_OUTPUT:
------------
100


Command was executed successfully

[Executed: 11/2/2016 11:46:44 PM] [Execution: 16ms]
```



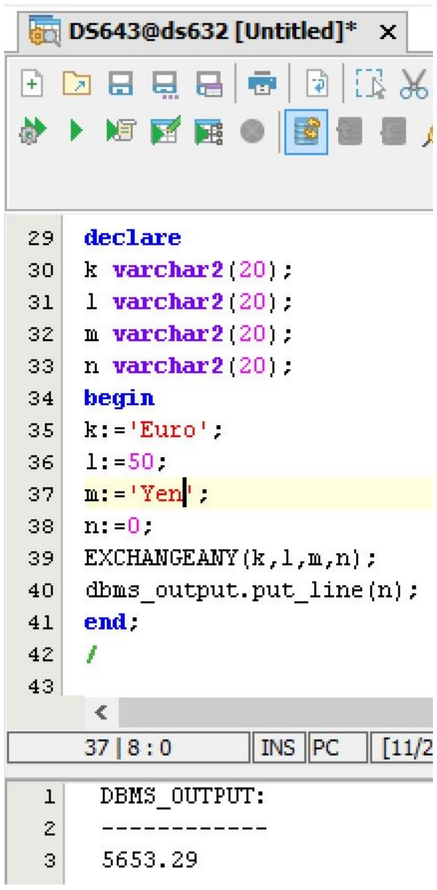
```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Dollar';
l:=100;
m:='Euro';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

```
DBMS_OUTPUT:
------------
89.66
```



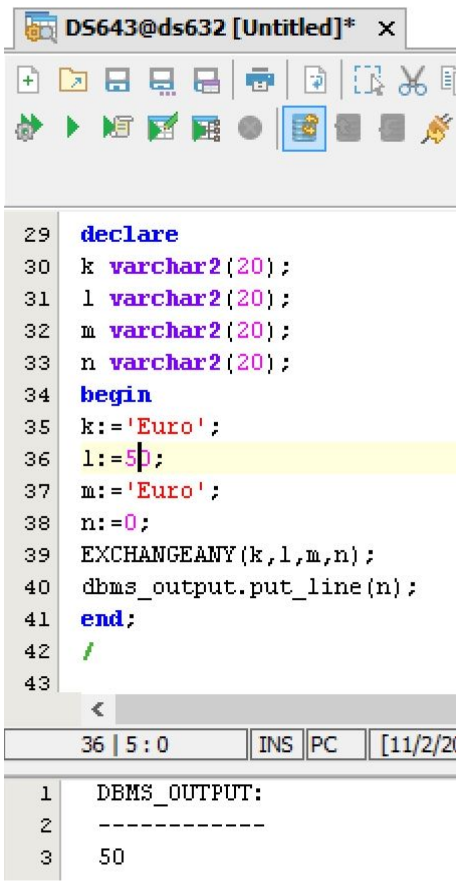
```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Yen';
l:=1000;
m:='Euro';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

```
DBMS_OUTPUT:
------------
8.85
```



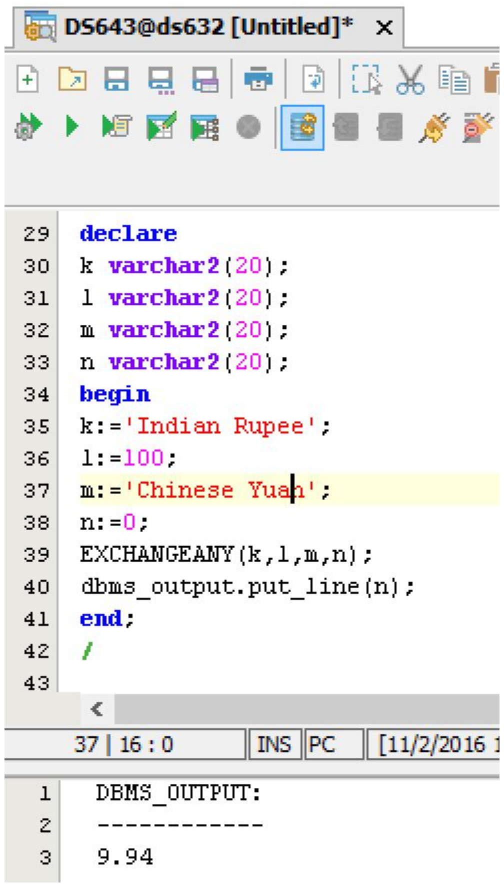
```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Euro';
l:=50;
m:='Yen';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

```
DBMS_OUTPUT:
------------
5653.29
```



```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Euro';
l:=50;
m:='Euro';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

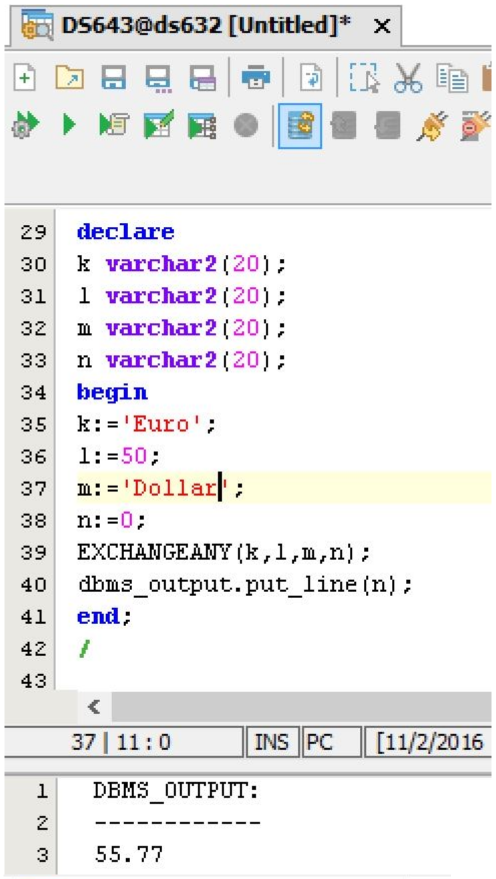
```
DBMS_OUTPUT:
------------
50
```


DS643@ds632 [Untitled]*

```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Indian Rupee';
l:=100;
m:='Chinese Yuan';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

37 | 16 : 0 INS PC [11/2/2016 1

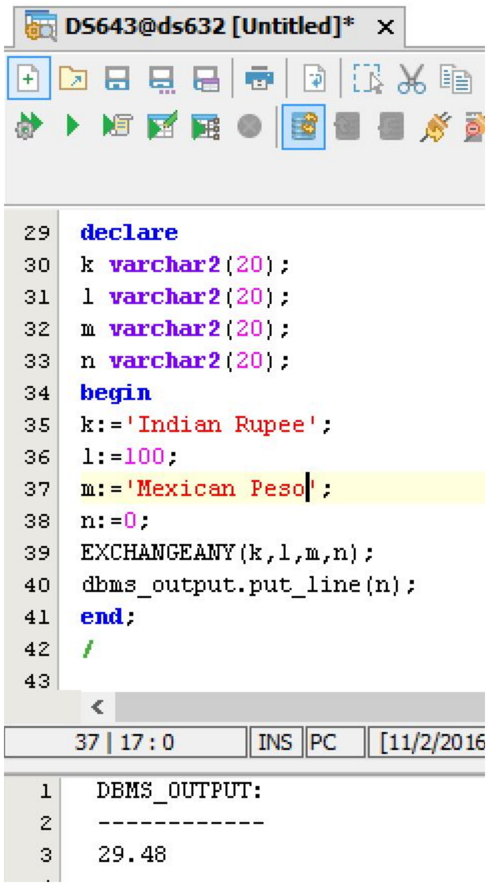
```
DBMS_OUTPUT:
------------
9.94
```


DS643@ds632 [Untitled]*

```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Euro';
l:=50;
m:='Dollar';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

37 | 11 : 0 INS PC [11/2/2016

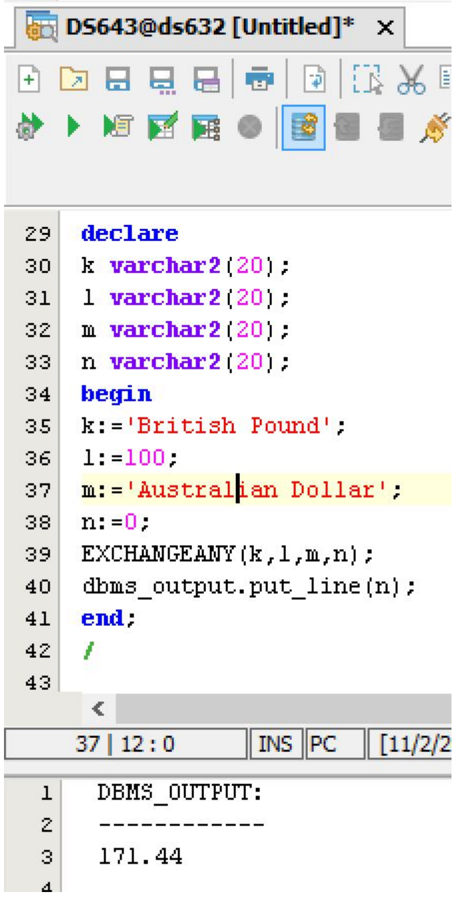
```
DBMS_OUTPUT:
------------
55.77
```


DS643@ds632 [Untitled]*

```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='Indian Rupee';
l:=100;
m:='Mexican Peso';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

37 | 17 : 0 INS PC [11/2/2016

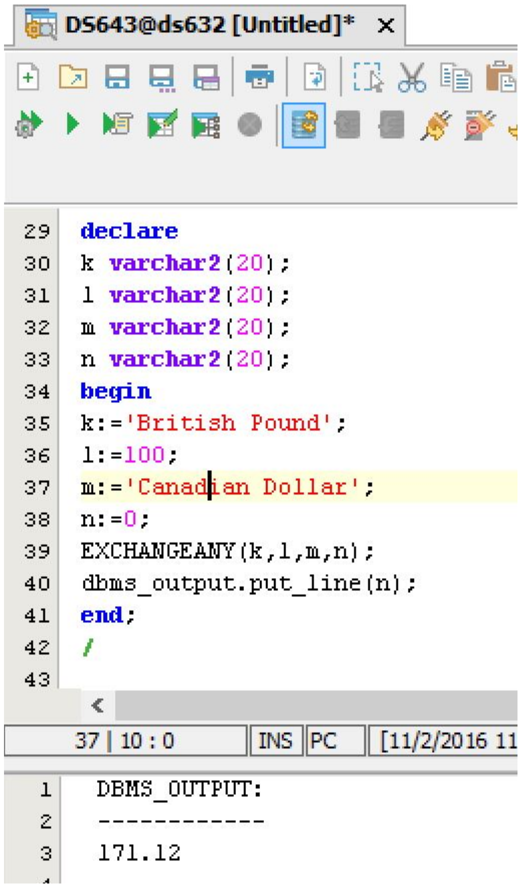
```
DBMS_OUTPUT:
------------
29.48
```


DS643@ds632 [Untitled]*

```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='British Pound';
l:=100;
m:='Australian Dollar';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

37 | 12 : 0 INS PC [11/2/2

```
DBMS_OUTPUT:
------------
171.44
```



```
declare
k varchar2(20);
l varchar2(20);
m varchar2(20);
n varchar2(20);
begin
k:='British Pound';
l:=100;
m:='Canadian Dollar';
n:=0;
EXCHANGEANY(k,l,m,n);
dbms_output.put_line(n);
end;
/
```

```
DBMS_OUTPUT:
------------
171.12
```