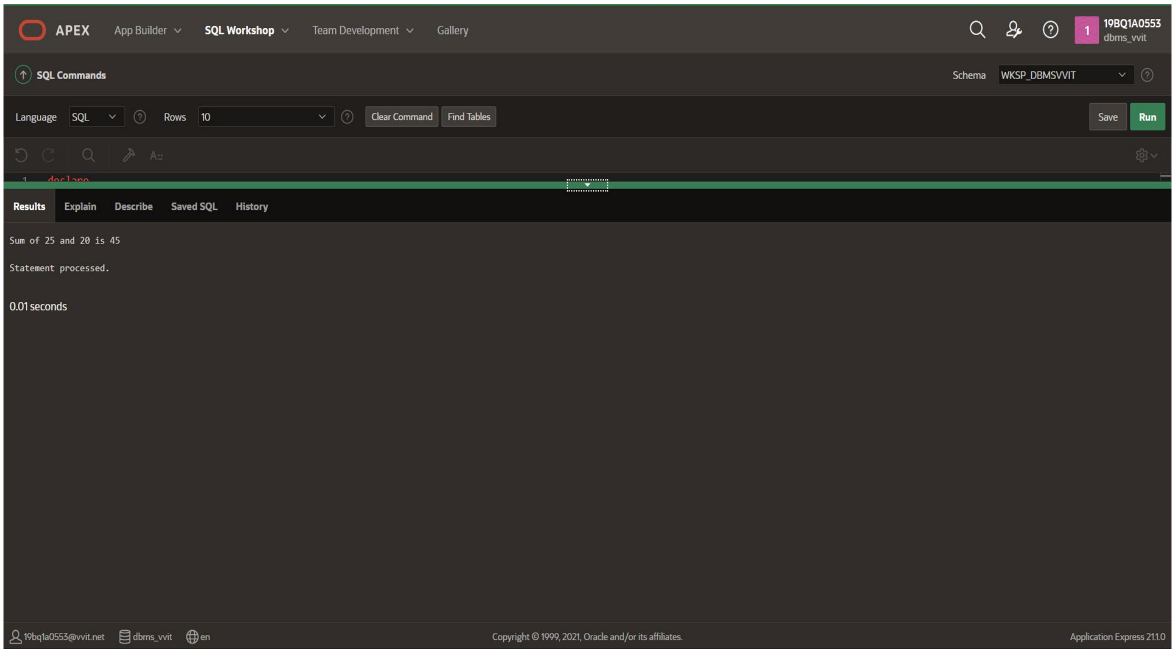
1.Write a PL/ SQL to add two numbers.

PL/SQL:-

```
declare
a number:=25;
b number:=20;
c number;
begin
c:=a+b;
dbms_output.put_line('Sum of '||a||' and '||b||' is '||c);
end;
```

Output:-



```
Sum of 25 and 20 is 45

Statement processed.
```

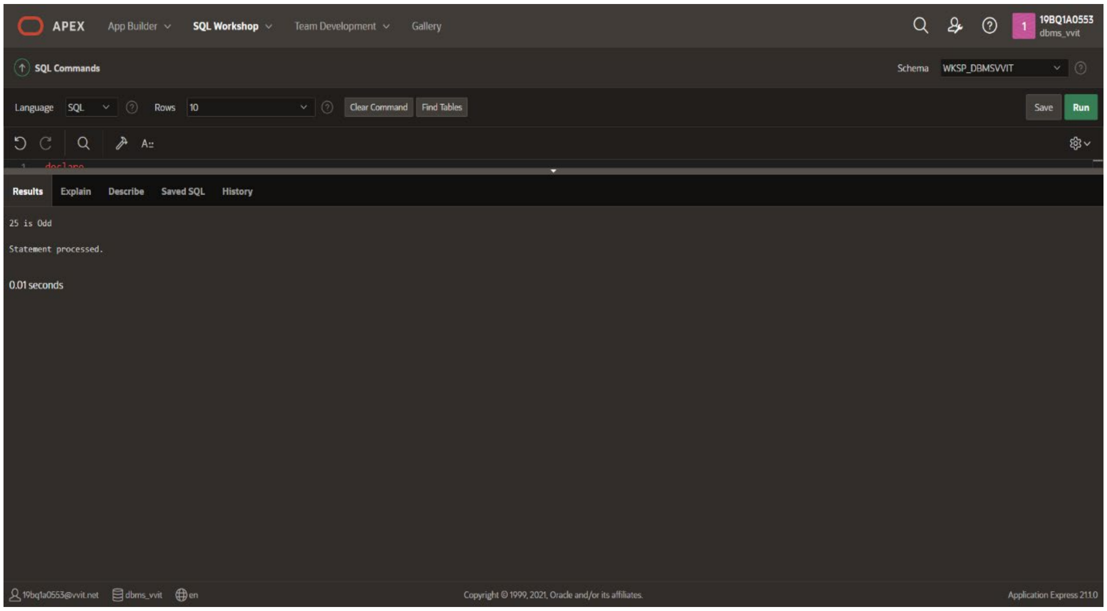
0.01 seconds

2. Write a PL/ SQL to check whether the given number is even or odd.

PL/ SQL:-

```
declare
num number:=25;
begin
if(mod(num,2)=0) then
dbms_output.put_line(num||' is Even');
else
dbms_output.put_line(num||' is Odd');
end if;
end;
```

Output:-



Schema WKSP_DBMSVVIT

Language SQL Rows 10

```
25 is Odd

Statement processed.
```

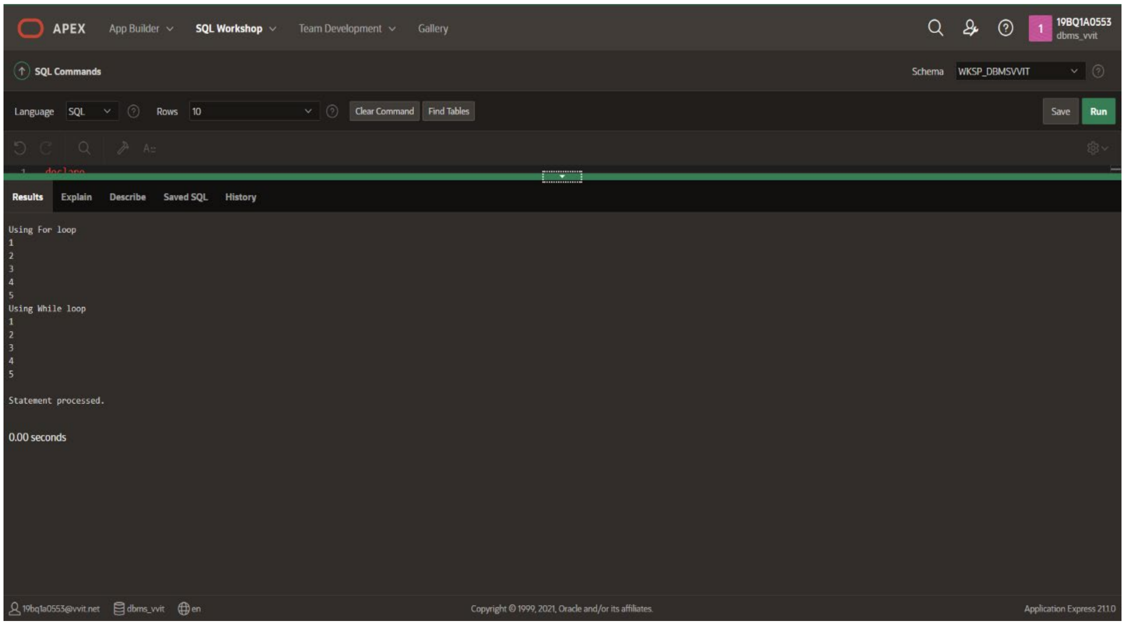
0.01 seconds

3. Write a PL/ SQL to generate natural number using various loops.

PL/ SQL:-

```
declare
a number:=5;
i number:=1;
begin
if(a>0) then
dbms_output.put_line('Using For loop');
for i in 1..a
loop
dbms_output.put_line(i);
end loop;
dbms_output.put_line('Using While loop');
while(i<=a)
loop
dbms_output.put_line(i);
i:=i+1;
end loop;
end if;
end;
```

Output:-



Schema WKSP_DBMSVVIT

Language SQL Rows 10

```
Using For loop
1
2
3
4
5
Using While loop
1
2
3
4
5

Statement processed.

0.00 seconds
```

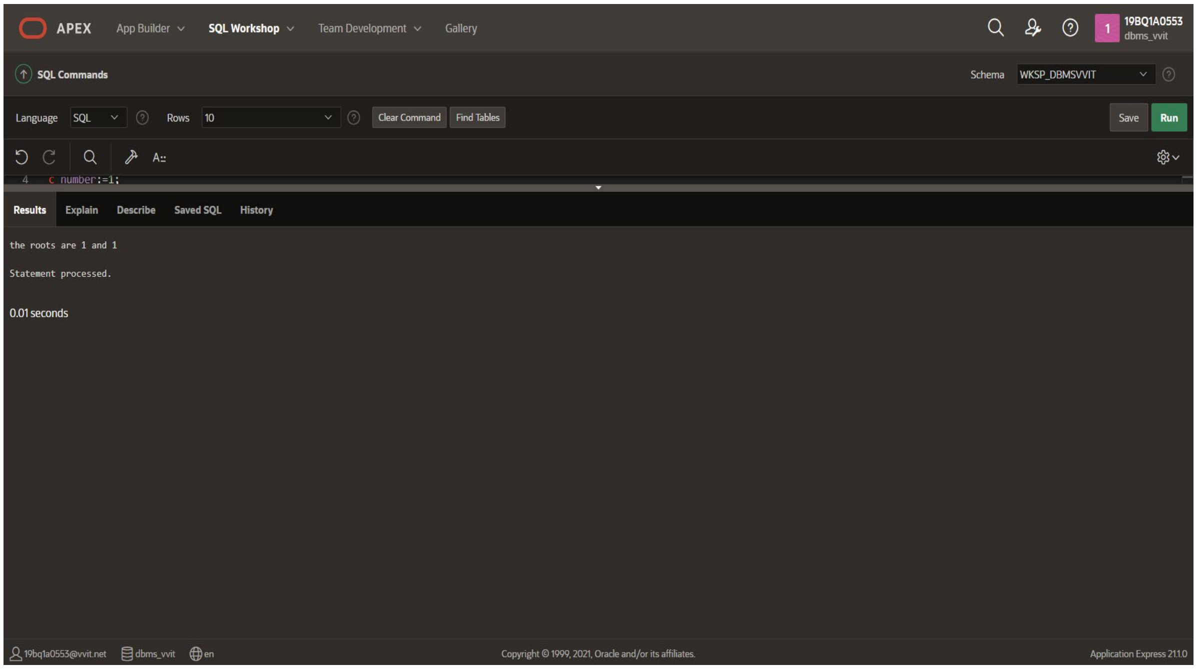
4. Write a PL/ SQL to find the roots of quadratic equation.

PL/ SQL:-

```
declare
a number:=1;
b number:=-2;
c number:=1;
const number:=0;
d number:=0;
r1 number:=0;
```

```
r2 number:=0;
begin
d:=b*b-4*a*c;
if(d>0) then
r1:=(-b+sqrt(d))/(2*a);
r2:=(-b-sqrt(d))/(2*a);
dbms_output.put_line('the roots are '
||r1|| ' and ' ||r2);
else
if(d=0)then
r1:=-b/(2*a);
dbms_output.put_line('the roots are '
||r1|| ' and ' ||r1);
else
r1:=-b/(2*a);
r2:=sqrt(-d)/(2*a);
dbms_output.put_line('the roots are '
||r1|| ' and ' ||'+i'||r2||r1||'-i'||r2);
end if;
end if;
end;
```

Output:-



```
c number:=1;
```

```
the roots are 1 and 1

Statement processed.
```

0.01 seconds

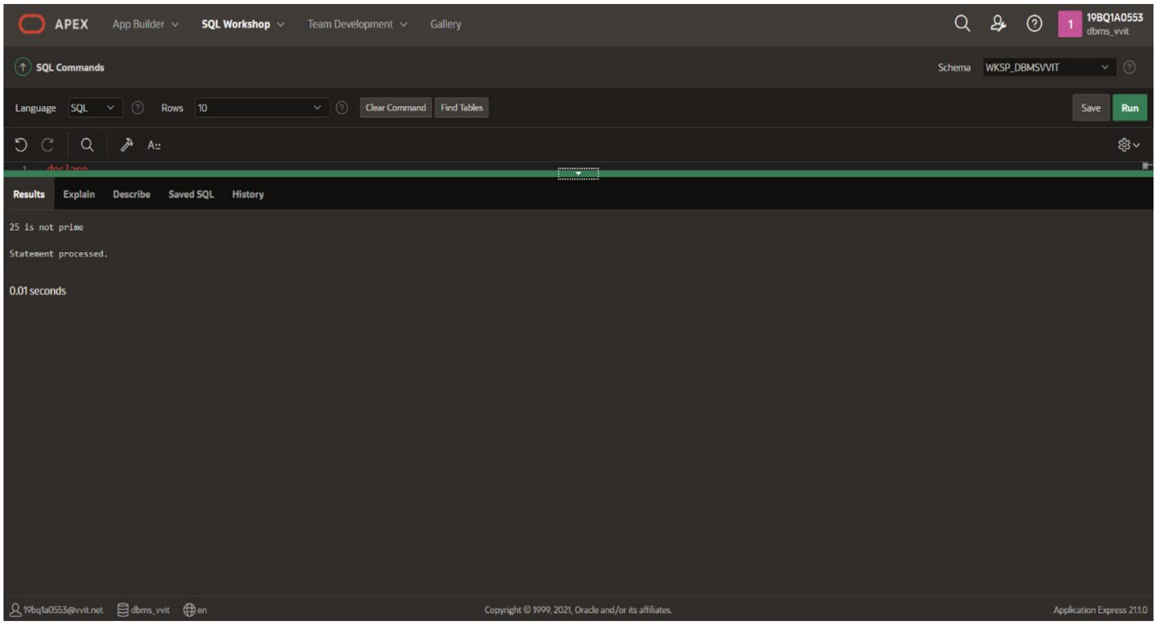
5. Write a PL/ SQL to check whether the given number is prime or not.

PL/ SQL:-

```
declare
n number :=25;
i number :=1;
cnt number :=0;
begin
for i in 1..n
loop
```

```
if(mod(n,i)=0) then
cnt :=cnt+1;
end if;
end loop;
if(cnt=2) then
dbms_output.put_line(n ||' is prime');
else
dbms_output.put_line(n ||' is not prime');
end if;
end;
```

Output:-



```
25 is not prime

Statement processed.

0.01 seconds
```

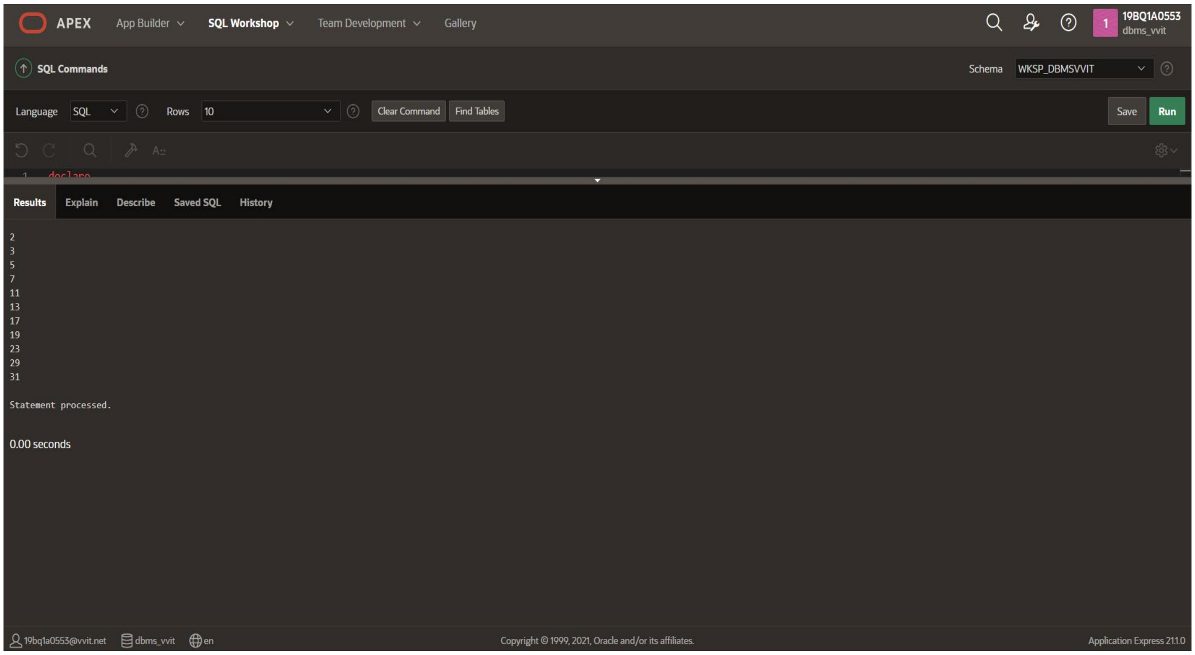
6. Write a PL/ SQL to generate prime numbers upto m.

PL/SQL:-

```
declare
m number :=35;
j number;
n number;
c number;
i number;
begin
j :=2;
while(j<=m)
loop
n :=j;
i :=1;
c :=0;
while(i<=n)
loop
if(mod(n,i)=0 OR (n=i)) then
c :=c+1;
end if;
```

```
i :=i+1;
end loop;
if(c=2) then
dbms_output.put_line(n);
end if;
j:=j+1;
end loop;
end;
```

Output:-



Schema WKSP_DBMSVVIT

Language SQL Rows 10

```
declare
```

```
2
3
5
7
11
13
17
19
23
29
31

Statement processed.
```

0.00 seconds

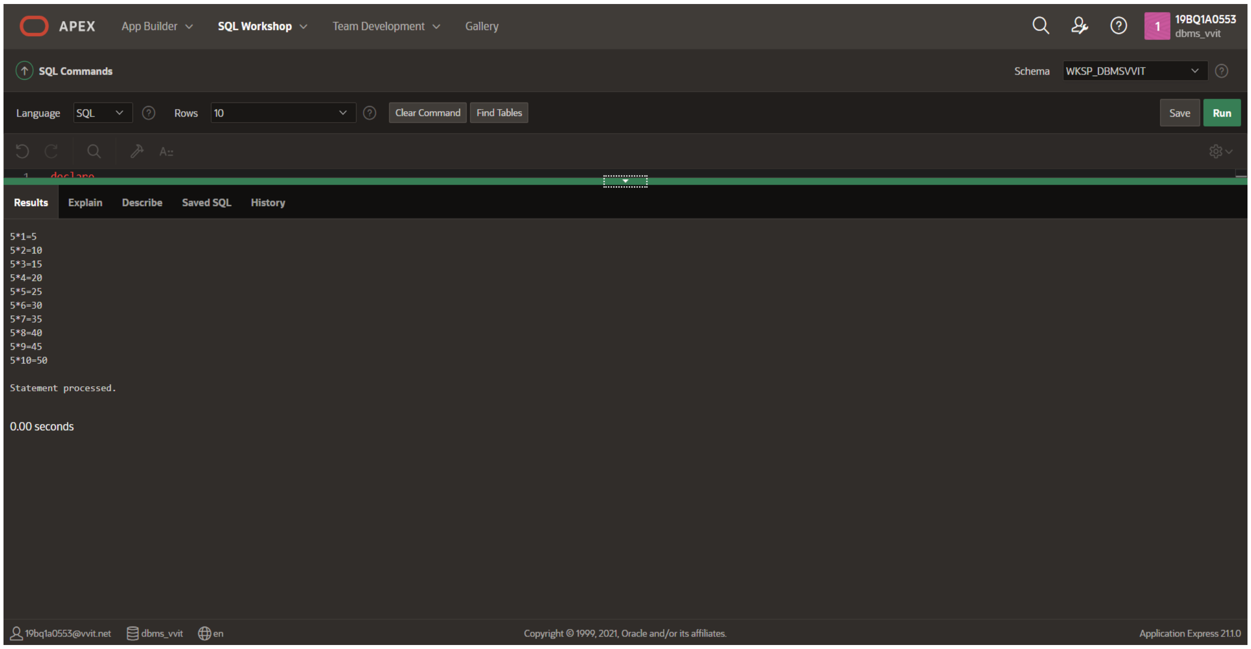
19bq1a0553@vvit.net dbms_vvit en

7. Write a PL/ SQL to generate mathematical product table for a given number.

PL/ SQL:-

```
declare
n number :=5;
i number :=1;
begin
for i in 1..10
loop
dbms_output.put_line(n ||'*'||i||'='||n*i);
end loop;
end;
```

Output:-



Schema WKSP_DBMSVVIT

Language SQL

Rows 10

```
declare
```

```
5*1=5
5*2=10
5*3=15
5*4=20
5*5=25
5*6=30
5*7=35
5*8=40
5*9=45
5*10=50

Statement processed.
```

0.00 seconds

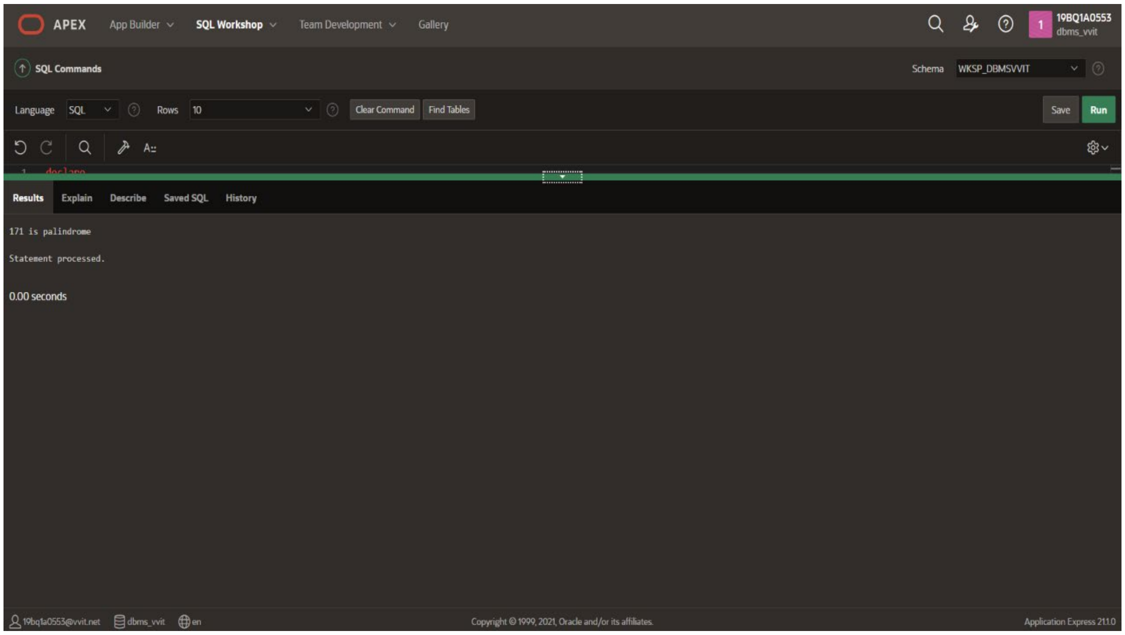
8. Write a PL/ SQL to check whether the given number is palindrome or not.

PL/ SQL:-

```
declare
n number :=171;
rem number :=0;
total number :=0;
k number :=0;
begin
k :=n;
while(n>0)
loop
rem :=mod(n,10);
total :=total*10+rem;
n :=trunc(n/10);
end loop;
if(total=k) then
dbms_output.put_line(k || ' is palindrome');
else
```

```
dbms_output.put_line(k || ' is not
palindrome');
end if;
end;
```

Output:-



```
171 is palindrome

Statement processed.
```

0.00 seconds

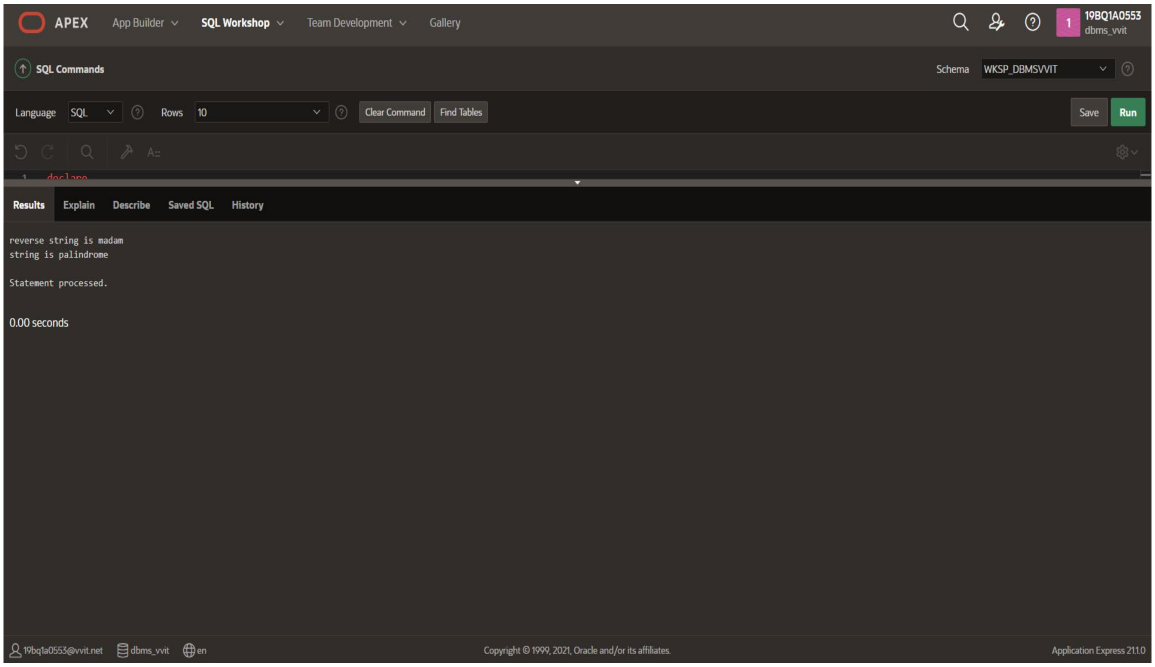
9. Write a PL/ SQL to check whether the given string is palindrome or not.

PL/ SQL:-

```
declare
g varchar2(20);
r varchar2(20);
i number(4);
```

```
begin
g:='madam';
for i in reverse 1..length(g)
loop
r:=r||substr(g,i,1);
end loop;
dbms_output.put_line('reverse string is '
||r);
if r=g then
dbms_output.put_line('string is
palindrome');
else
dbms_output.put_line('string is not
palindrome');
end if;
end;
```

Output:-



```
declare
```

```
reverse string is madam
string is palindrome

Statement processed.
```

0.00 seconds