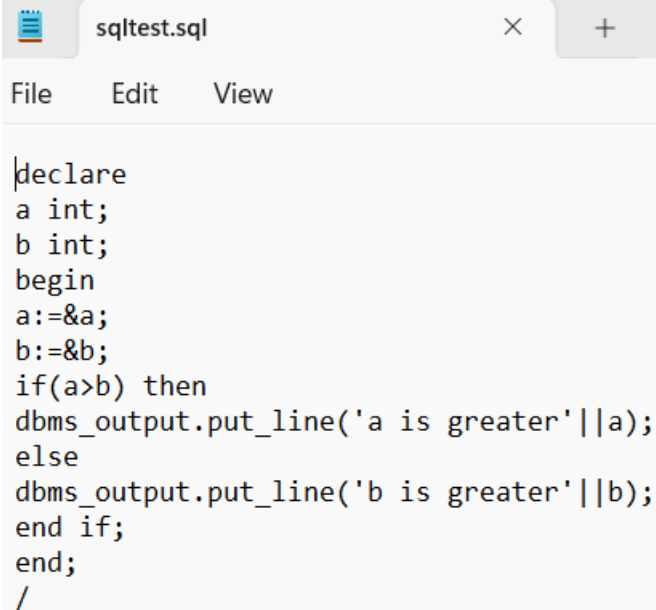


SIMPLE IF-ELSE

```
SQL> set serveroutput on
SQL> edit sqltest
```



The screenshot shows a web-based SQL editor interface. At the top, there is a tab labeled 'sqltest.sql' with a close button (X) and a plus sign (+) for new tabs. Below the tab is a menu bar with 'File', 'Edit', and 'View' options. The main area contains the following PL/SQL code:

```
declare
a int;
b int;
begin
a:=&a;
b:=&b;
if(a>b) then
dbms_output.put_line('a is greater' || a);
else
dbms_output.put_line('b is greater' || b);
end if;
end;
/
```

```
SQL> set serveroutput on
SQL> edit sqltest

SQL> @sqltest
Enter value for a: 56
old   5: a:=&a;
new   5: a:=56;
Enter value for b: 75
old   6: b:=&b;
new   6: b:=75;
b is greater75

PL/SQL procedure successfully completed.

SQL> |
```

FETCHING DATA FROM EMPLOYEE TABLE

```
SQL> select * from employee;
```

ID	NAME	DOJ	SALARY	DID
101	jack	07-DEC-94	15655	1
102	kay	05-AUG-96	18500	3
103	lisa	14-OCT-91	25000	5
104	ray	21-NOV-97	11000	1
105	alex	28-SEP-96	16000	2

```
SQL> set serveroutput on
```

```
SQL> edit sqltest
```

```
sqltest.sql
File Edit View

declare
  eid number :=103;
  esal number;
begin
  select salary into esal from employee where id=eid;
  dbms_output.put_line('The salary of employee is: '||esal);
end;
/
```

```
SQL> set serveroutput on
```

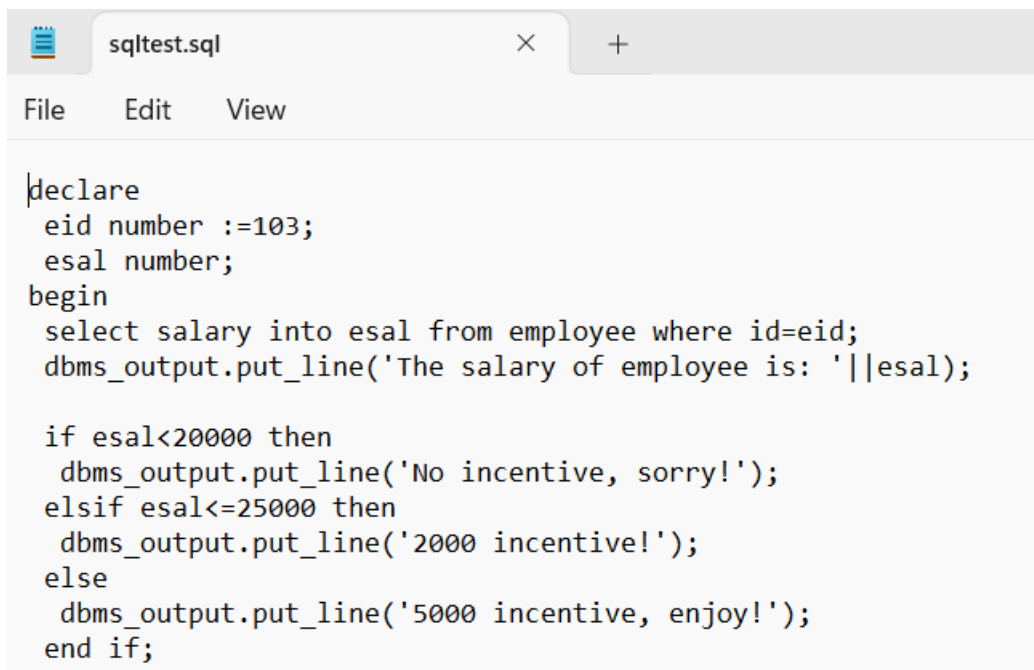
```
SQL> edit sqltest
```

```
SQL> @sqltest
```

```
The salary of employee is: 25000
```

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

```
SQL> |
```



```
declare
  eid number :=103;
  esal number;
begin
  select salary into esal from employee where id=eid;
  dbms_output.put_line('The salary of employee is: '||esal);

  if esal<20000 then
    dbms_output.put_line('No incentive, sorry!');
  elsif esal<=25000 then
    dbms_output.put_line('2000 incentive!');
  else
    dbms_output.put_line('5000 incentive, enjoy!');
  end if;
```

```
SQL> edit sqltest

SQL> @sqltest
The salary of employee is: 25000
2000 incentive!
THANK YOU!

PL/SQL procedure successfully completed.
```

UPDATE SALARY OF EMPLOYEE TABLE

```
SQL> select * from employee;
```

ID	NAME	DOJ	SALARY	DID
101	jack	07-DEC-94	15655	1
102	kay	05-AUG-96	18500	3
103	lisa	14-OCT-91	25000	5
104	ray	21-NOV-97	11000	1
105	alex	28-SEP-96	16000	2

```
sqltest.sql × +  
File Edit View  
  
declare  
sal number:=&sal;  
eid number:=102;  
begin  
  update employee set salary = sal where id=eid;  
  dbms_output.put_line('Salary updated successfully');  
end;  
/
```

```
SQL> @sqltest  
Enter value for sal: 23000  
old 2: sal number:=&sal;  
new 2: sal number:=23000;  
Salary updated successfully  
  
PL/SQL procedure successfully completed.  
  
SQL> select * from employee;
```

ID	NAME	DOJ	SALARY	DID
101	jack	07-DEC-94	15655	1
102	kay	05-AUG-96	23000	3
103	lisa	14-OCT-91	25000	5
104	ray	21-NOV-97	11000	1
105	alex	28-SEP-96	16000	2

```
SQL> |
```

FETCHING DATA FROM EMPLOYEE WITH LIKE OPERATOR

```
SQL> desc employee;
```

Name	Null?	Type
ID	NOT NULL	NUMBER
NAME		VARCHAR2(20)
DOJ		DATE
SALARY		NUMBER
DID		NUMBER

```
SQL> select * from employee;
```

ID	NAME	DOJ	SALARY	DID
101	jack	07-DEC-94	15655	1
102	kay	05-AUG-96	23000	3
103	lisa	14-OCT-91	25000	5
104	ray	21-NOV-97	11000	1
105	alex	28-SEP-96	16000	2

```
declare
  ename varchar2(20);

begin
  select name into ename from employee where name like '%ac%';
  dbms_output.put_line('The employee name is '||ename);
end;
/
```

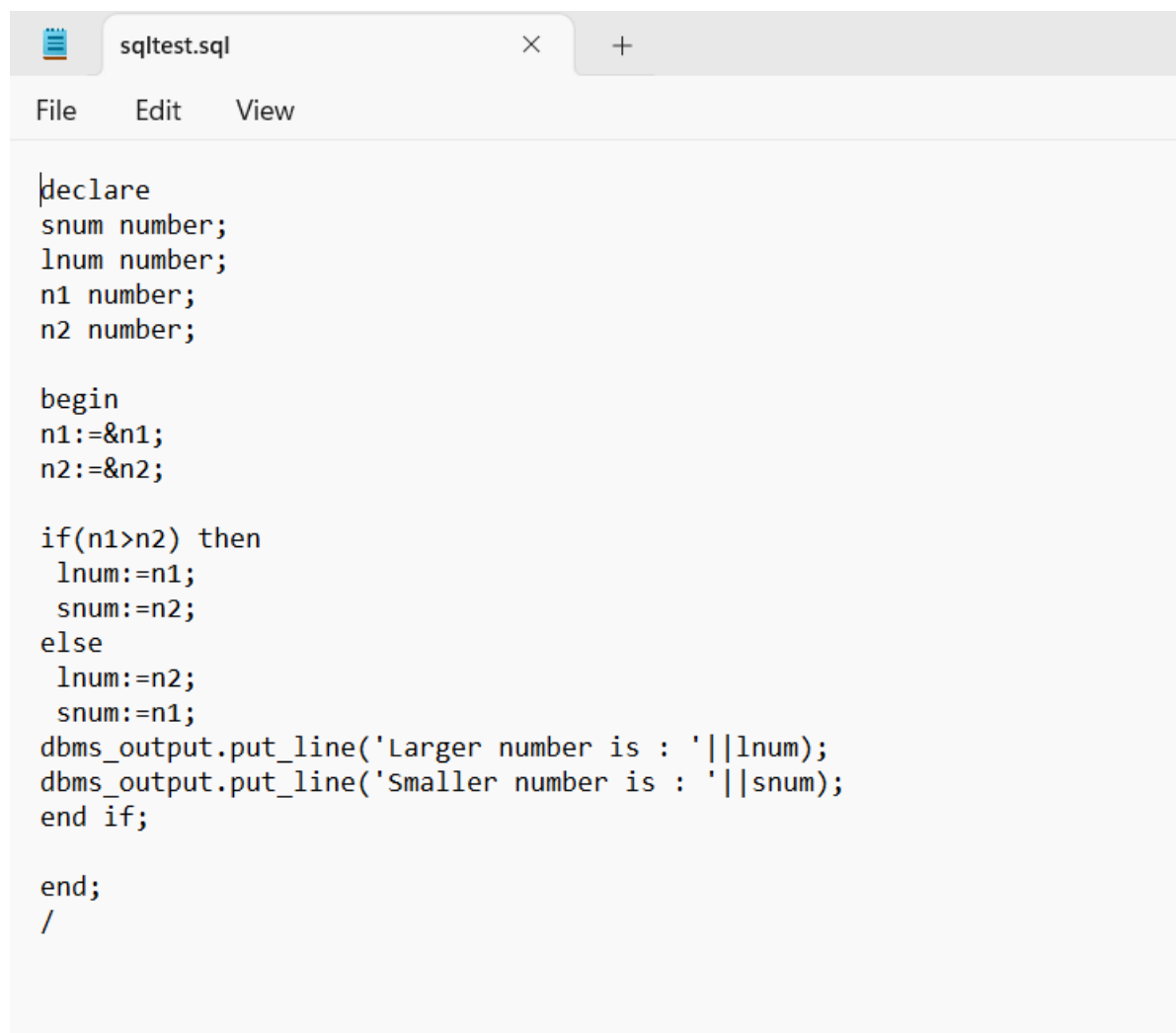
```
SQL> edit sqltest
```

```
SQL> @sqltest
```

```
The employee name is jack
```

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

Write a PL/SQL program to arrange the number of two variable in such a way that the small number will store in num_small variable and large number will store in num_large variable.



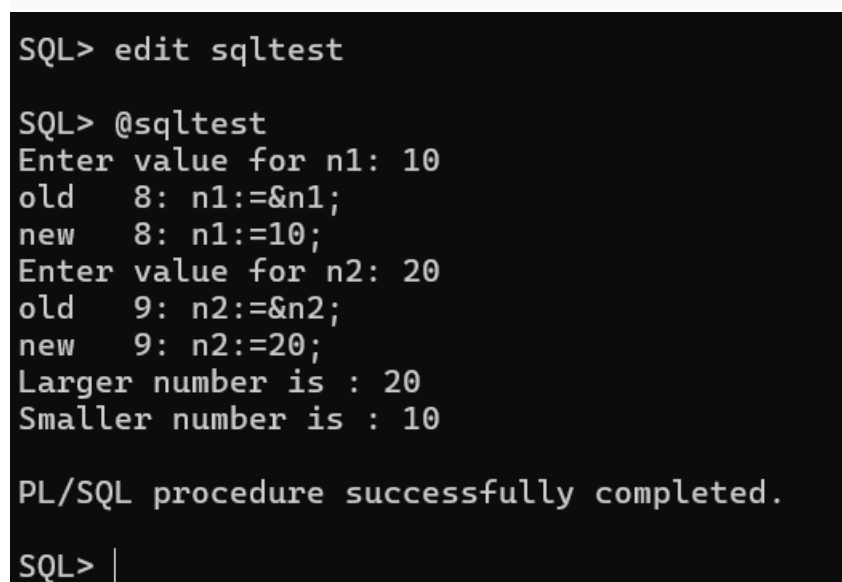
```
sqltest.sql
File Edit View

declare
snum number;
lnum number;
n1 number;
n2 number;

begin
n1:=&n1;
n2:=&n2;

if(n1>n2) then
    lnum:=n1;
    snum:=n2;
else
    lnum:=n2;
    snum:=n1;
dbms_output.put_line('Larger number is : '||lnum);
dbms_output.put_line('Smaller number is : '||snum);
end if;

end;
/
```



```
SQL> edit sqltest

SQL> @sqltest
Enter value for n1: 10
old   8: n1:=&n1;
new   8: n1:=10;
Enter value for n2: 20
old   9: n2:=&n2;
new   9: n2:=20;
Larger number is : 20
Smaller number is : 10

PL/SQL procedure successfully completed.

SQL> |
```

Write a PL/SQL program to count number of employees in a specific department and check whether this department have any vacancies or not. If any vacancies, how many vacancies are in that department.

```
SQL> select * from employee;
```

ID	NAME	DOJ	SALARY	DID
101	jack	07-DEC-94	15655	1
102	kay	05-AUG-96	23000	3
103	lisa	14-OCT-91	25000	5
104	ray	21-NOV-97	11000	1
105	alex	28-SEP-96	16000	2



sqltest.sql



File Edit View

```
declare
permitted number;
etotal number;

begin
permitted:=&permitted;
select count(*) into etotal from employee where did=1 or did=2;
dbms_output.put_line('Total Number of employees permitted :'||permitted);
dbms_output.put_line('Total Number of employees present :'||etotal);
if(etotal=permitted) then
dbms_output.put_line('NO VACANCIES');
else
dbms_output.put_line('VACANCIES PRESENT : '||(permitted-etotal));
end if;

end;
/
```

```
SQL> set serveroutput on
```

```
SQL> edit sqltest
```

```
SQL> @sqltest
```

```
Enter value for permitted: 5
```

```
old 6: permitted:=&permitted;
```

```
new 6: permitted:=5;
```

```
Total Number of employees permitted :5
```

```
Total Number of employees present :3
```

```
VACANCIES PRESENT : 2
```

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

```
SQL> |
```

```

SQL> @sqltest
Enter value for permitted: 3
old 6: permitted:=&permitted;
new 6: permitted:=3;
Total Number of employees permitted :3
Total Number of employees present :3
NO VACANCIES

PL/SQL procedure successfully completed.

SQL> |

```

Write a PL/SQL program to display the employee IDs, names, job titles, hire dates, and salaries of all employees.

```

create or replace procedure info
as
begin
    for emp in (
        select id as eid, name as ename, salary as esal from employee
    )
    loop
        dbms_output.put_line('Employee ID: '||emp.eid||', Name: '||emp.ename||', Salary: '||emp.esal);
    end loop;
end;
/

```

```

SQL> set serveroutput on
SQL> edit sqltest

SQL> @sqltest

Procedure created.

SQL> exec info;
Employee ID: 101, Name: jack, Salary: 15655
Employee ID: 102, Name: kay, Salary: 23000
Employee ID: 103, Name: lisa, Salary: 25000
Employee ID: 104, Name: ray, Salary: 11000
Employee ID: 105, Name: alex, Salary: 16000

PL/SQL procedure successfully completed.

```


Write a PL/SQL program to display the employee IDs, names, and department names of all employees.

```
create or replace procedure info
as
begin
    for emp in (
        select e.id as eid,e.name as ename, e.salary as esal,d.dname as edept from employee e inner join dept d on e.did=d.did
    )
    loop
        dbms_output.put_line('Employee ID: '||emp.eid||', Name: '||emp.ename||', Salary: '||emp.esal||', Department: '||emp.edept);
    end loop;
end;
/
```

```
SQL> set serveroutput on
SQL> edit sqltest
```

```
SQL> @sqltest
```

Procedure created.

```
SQL> exec info;
Employee ID: 101, Name: jack, Salary: 15655, Department: cse
Employee ID: 104, Name: ray, Salary: 11000, Department: cse
Employee ID: 105, Name: alex, Salary: 16000, Department: it
Employee ID: 102, Name: kay, Salary: 23000, Department: ece
Employee ID: 103, Name: lisa, Salary: 25000, Department: ft
```

PL/SQL procedure successfully completed.

```
SQL> select * from employee;
```

ID	NAME	DOJ	SALARY	DID
101	jack	07-DEC-94	15655	1
102	kay	05-AUG-96	23000	3
103	lisa	14-OCT-91	25000	5
104	ray	21-NOV-97	11000	1
105	alex	28-SEP-96	16000	2

```
SQL> select * from dept;
```

DID	DNAME	DLOC
1	cse	admin
2	it	admin
3	ece	workshop
4	eee	workshop
5	ft	aero

```
SQL> |
```