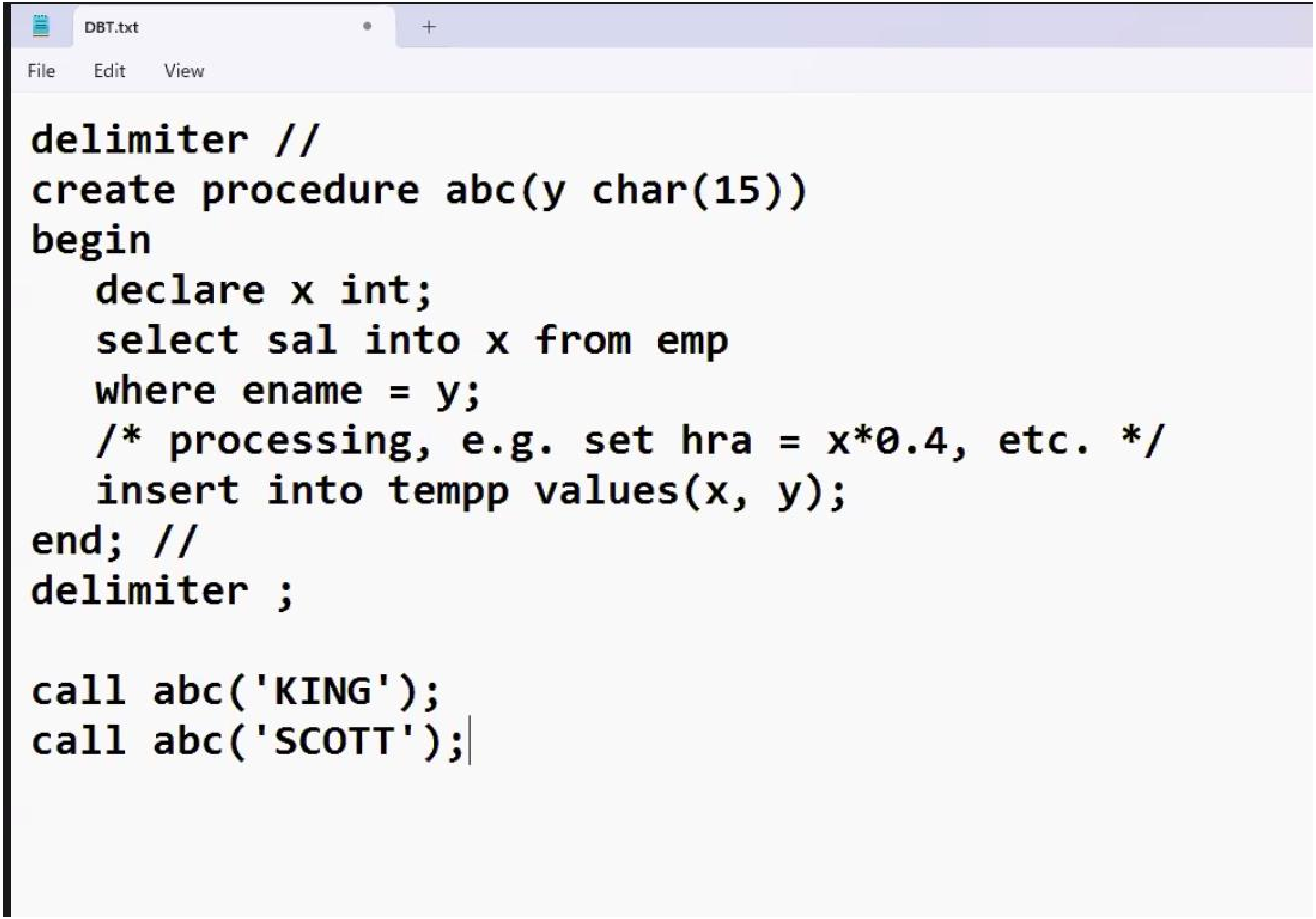
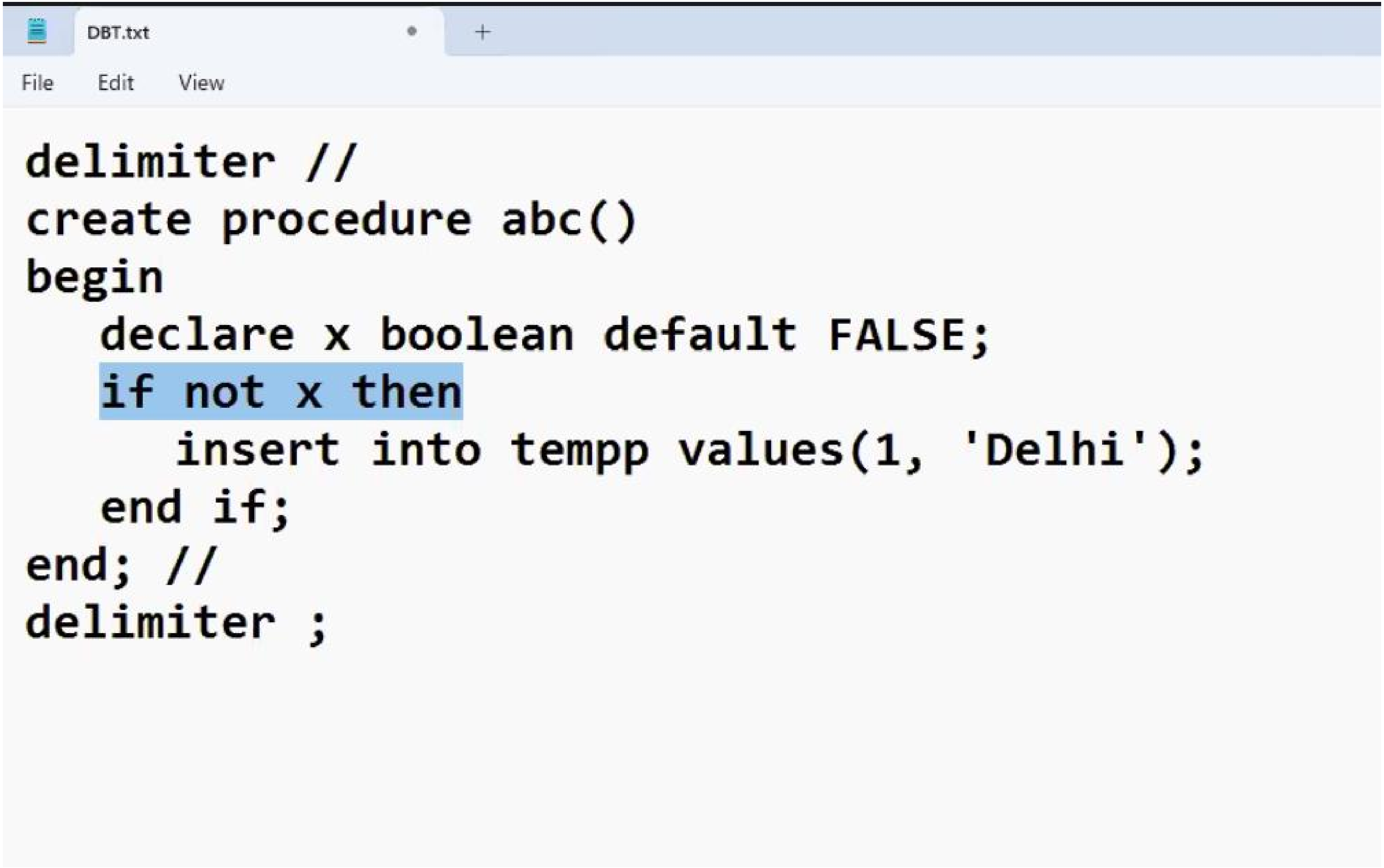
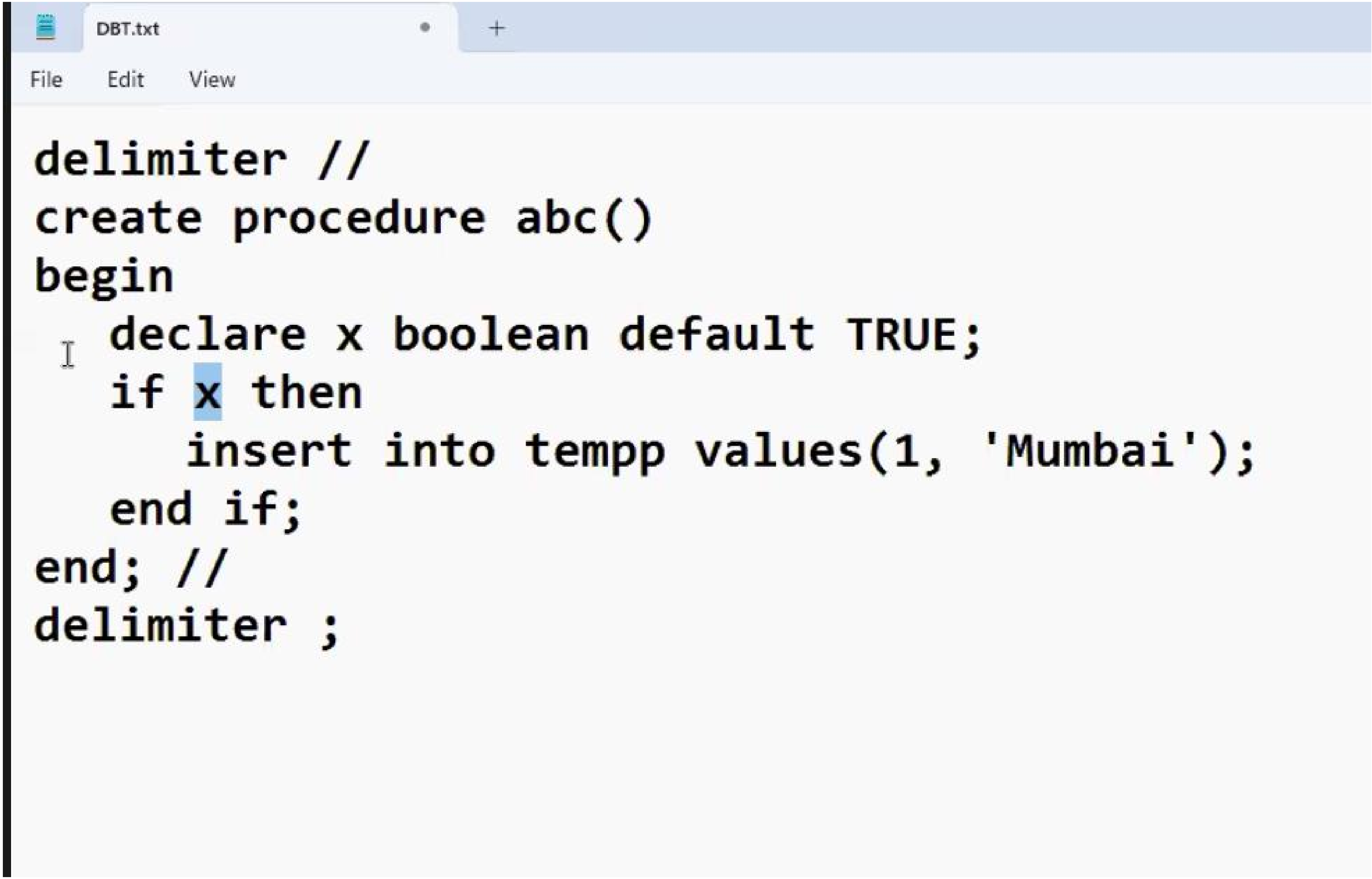


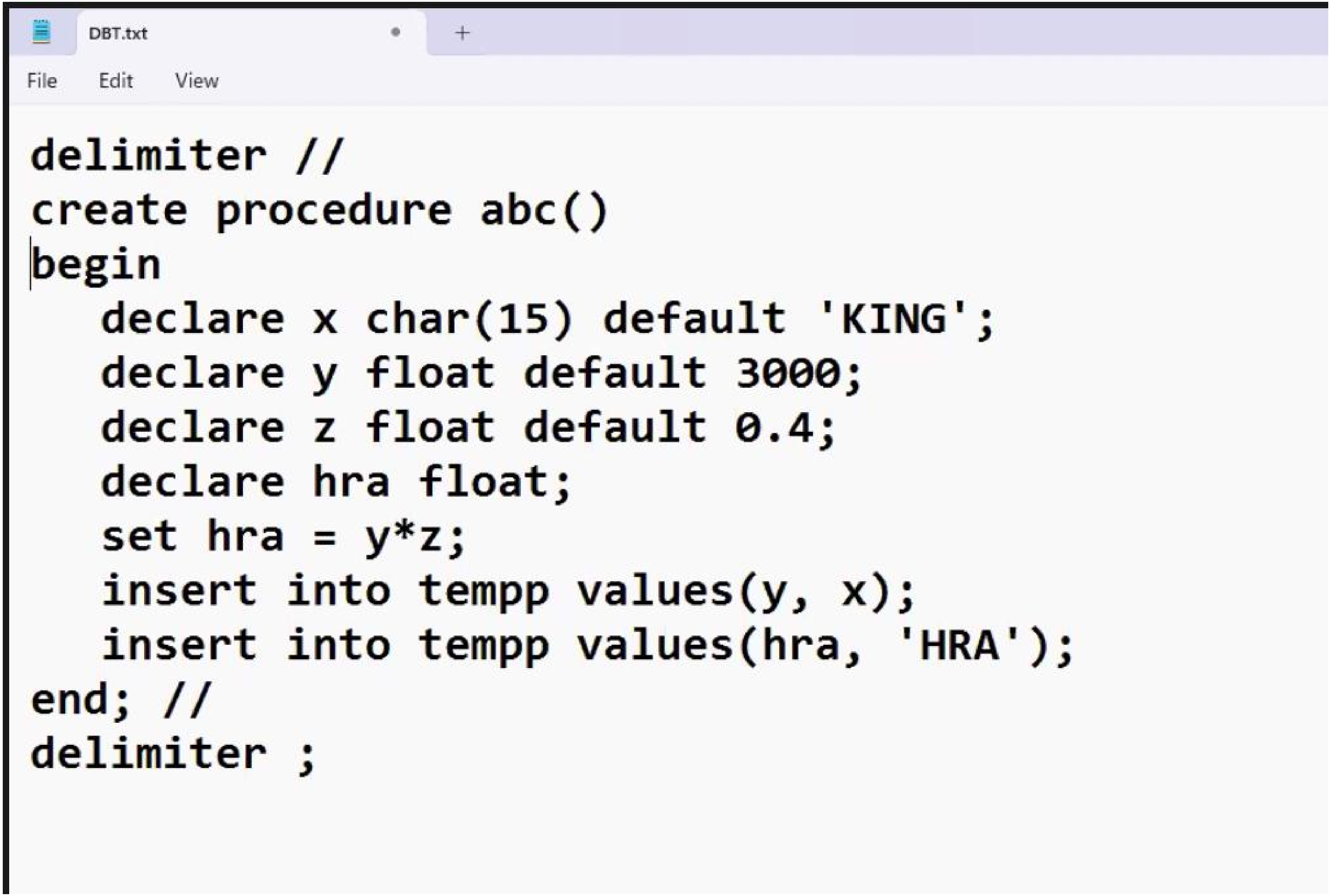
```
delimiter //
create procedure abc(y char(15))
begin
   declare x int;
   select sal into x from emp
   where ename = y;
   /* processing, e.g. set hra = x*0.4, etc. */
   insert into tempp values(x, y);
end; //
delimiter ;

call abc('KING');
call abc('SCOTT');
```



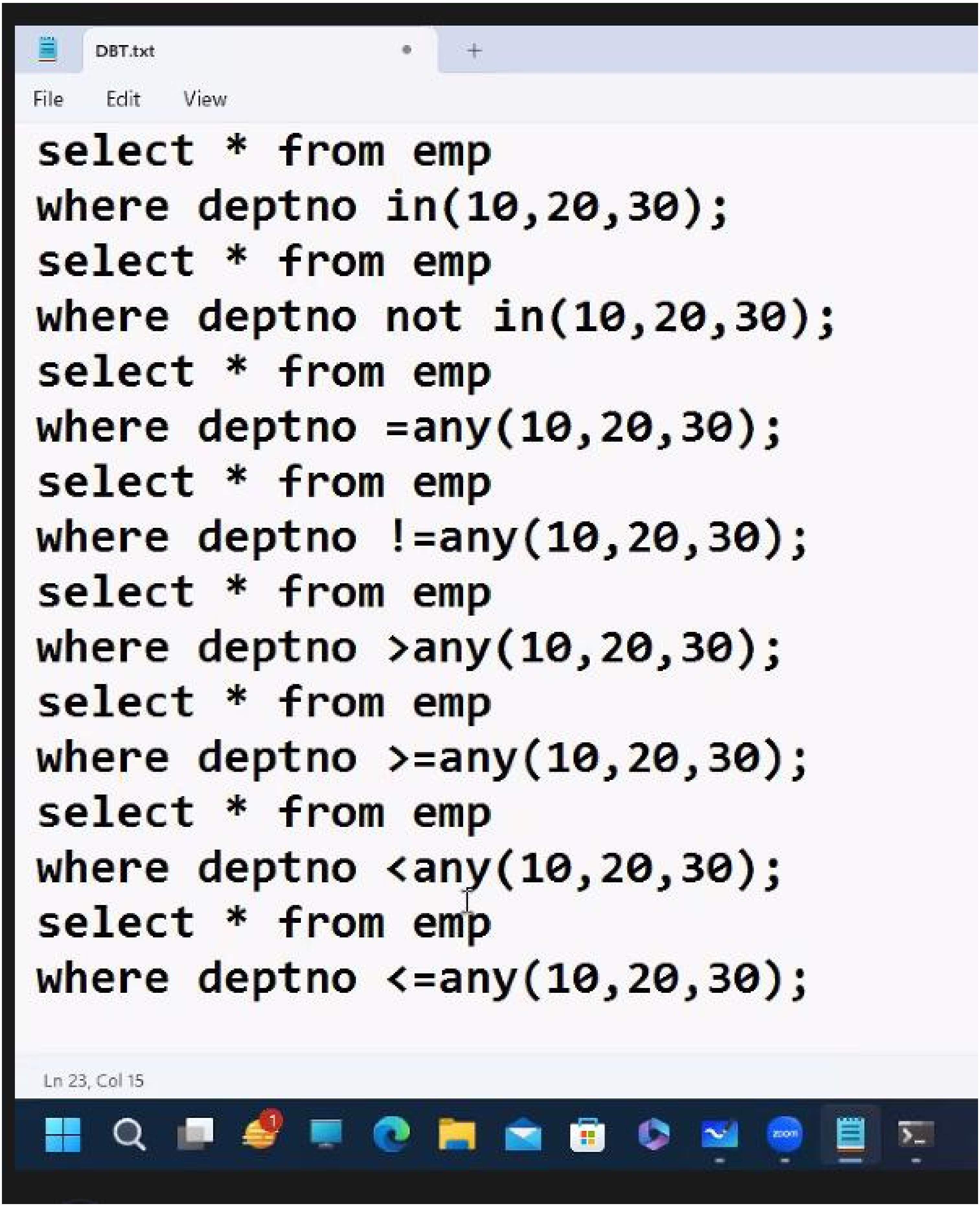
```
delimiter //
create procedure abc()
begin
   declare x boolean default FALSE;
   if not x then
      insert into tempp values(1, 'Delhi');
   end if;
end; //
delimiter ;
```



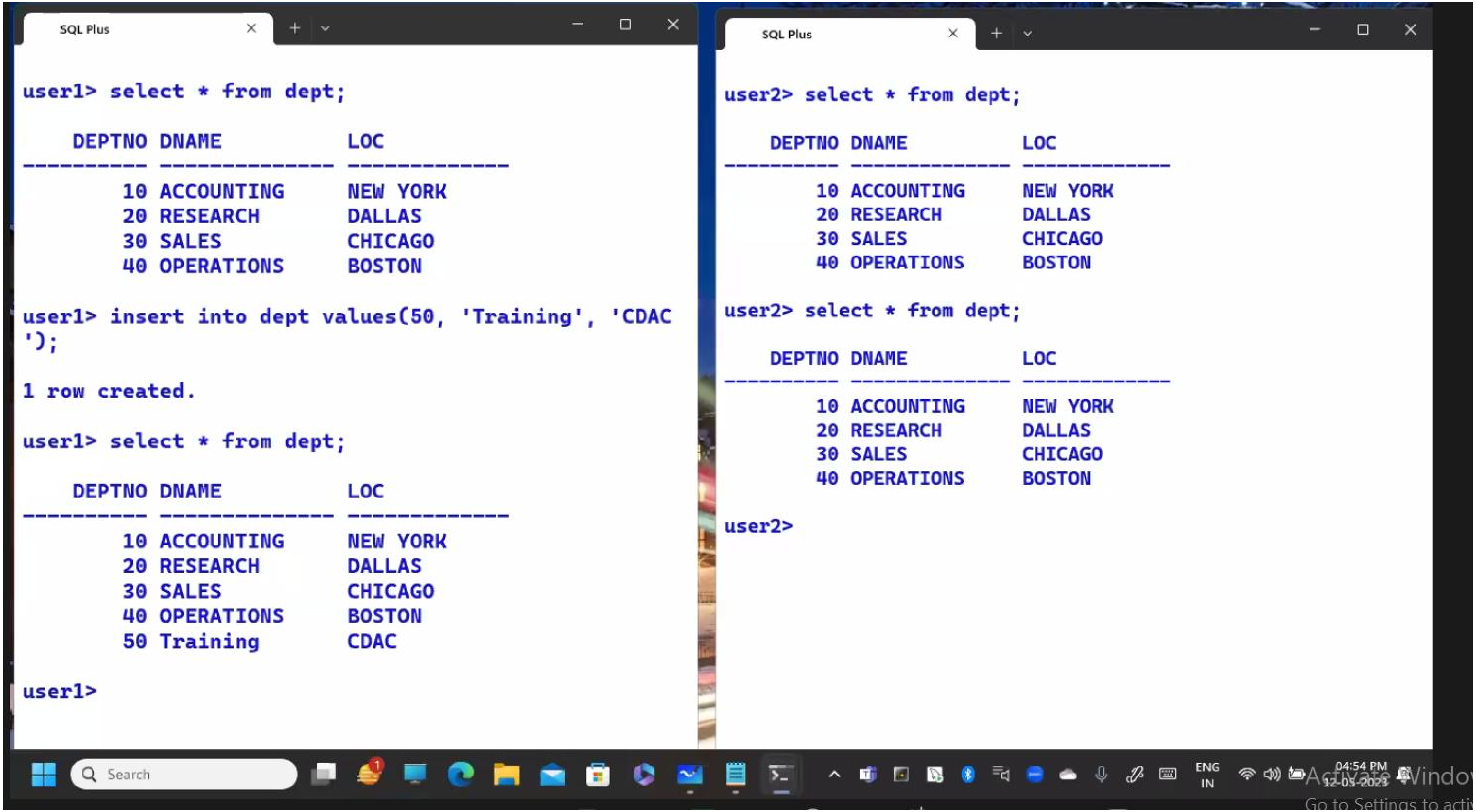
```
delimiter //
create procedure abc()
begin
   declare x boolean default TRUE;
   if x then
      insert into tempp values(1, 'Mumbai');
   end if;
end; //
delimiter ;
```



```
delimiter //
create procedure abc()
begin
   declare x char(15) default 'KING';
   declare y float default 3000;
   declare z float default 0.4;
   declare hra float;
   set hra = y*z;
   insert into tempp values(y, x);
   insert into tempp values(hra, 'HRA');
end; //
delimiter ;
```

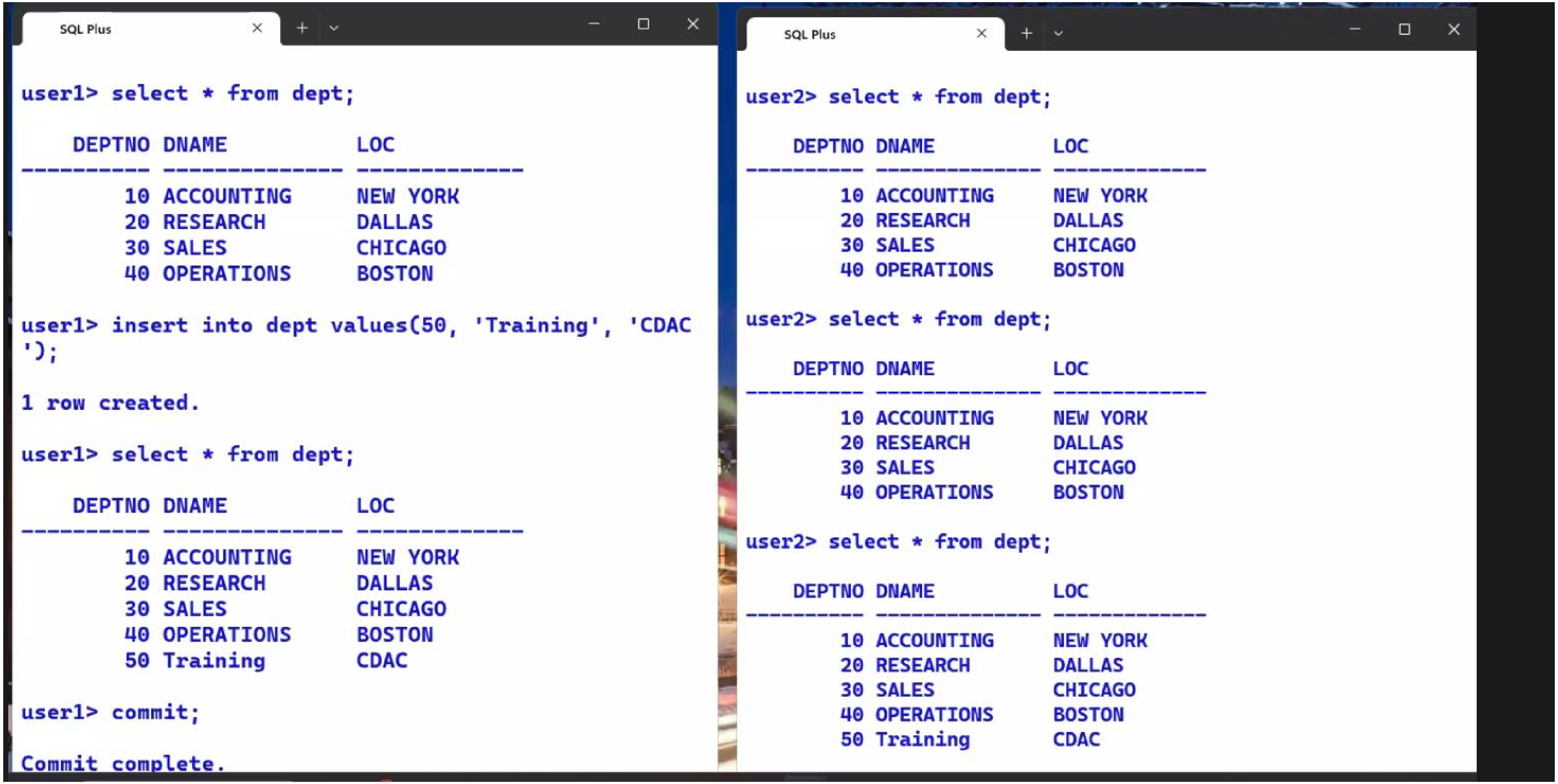


```
select * from emp
where deptno in(10,20,30);
select * from emp
where deptno not in(10,20,30);
select * from emp
where deptno =any(10,20,30);
select * from emp
where deptno !=any(10,20,30);
select * from emp
where deptno >any(10,20,30);
select * from emp
where deptno >=any(10,20,30);
select * from emp
where deptno <any(10,20,30);
select * from emp
where deptno <=any(10,20,30);
```



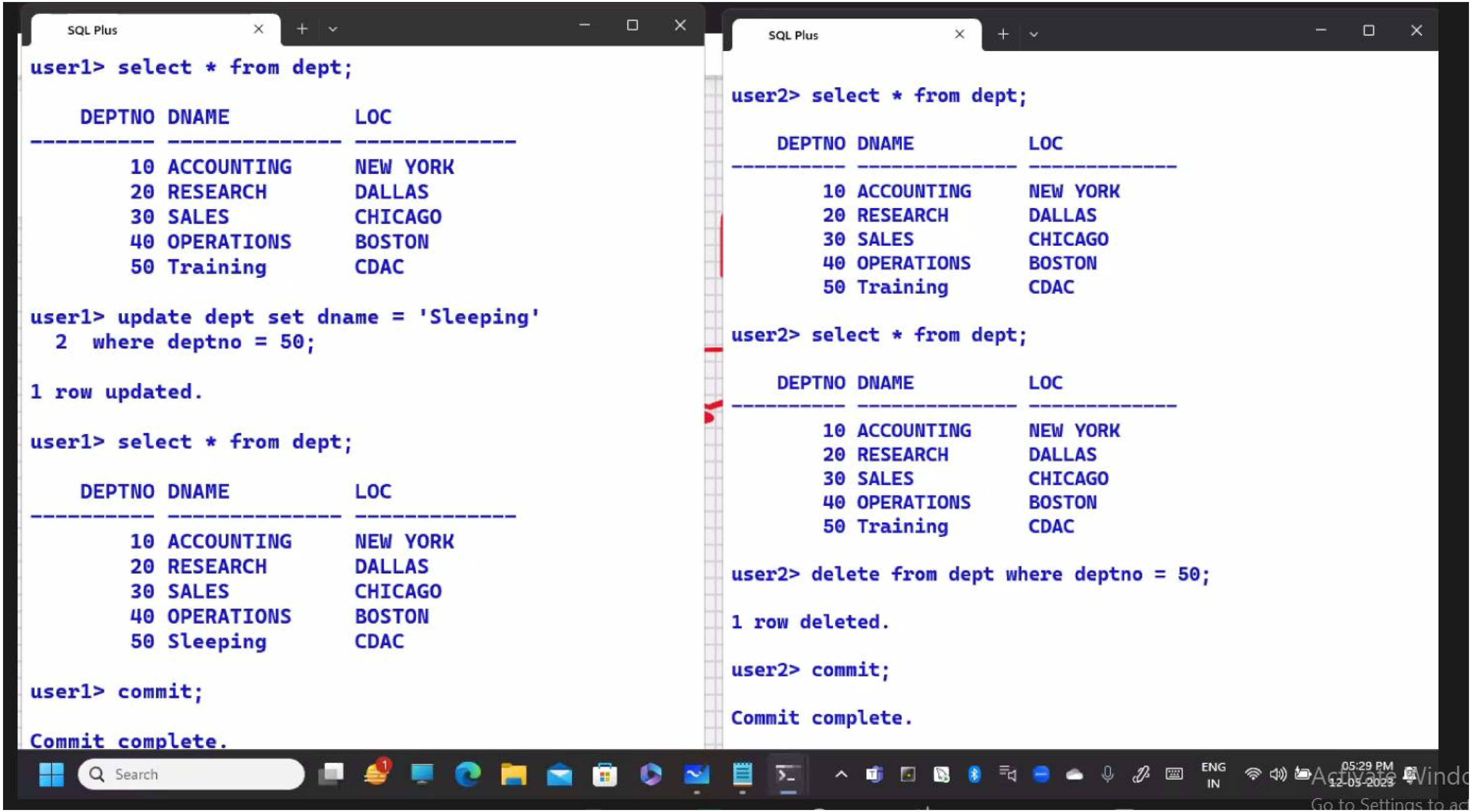
```
user1> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

user1> insert into dept values(50, 'Training', 'CDAC
');

1 row created.

user1> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
        50 Training       CDAC

user1>
```

```
user2> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

user2> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

user2>
```



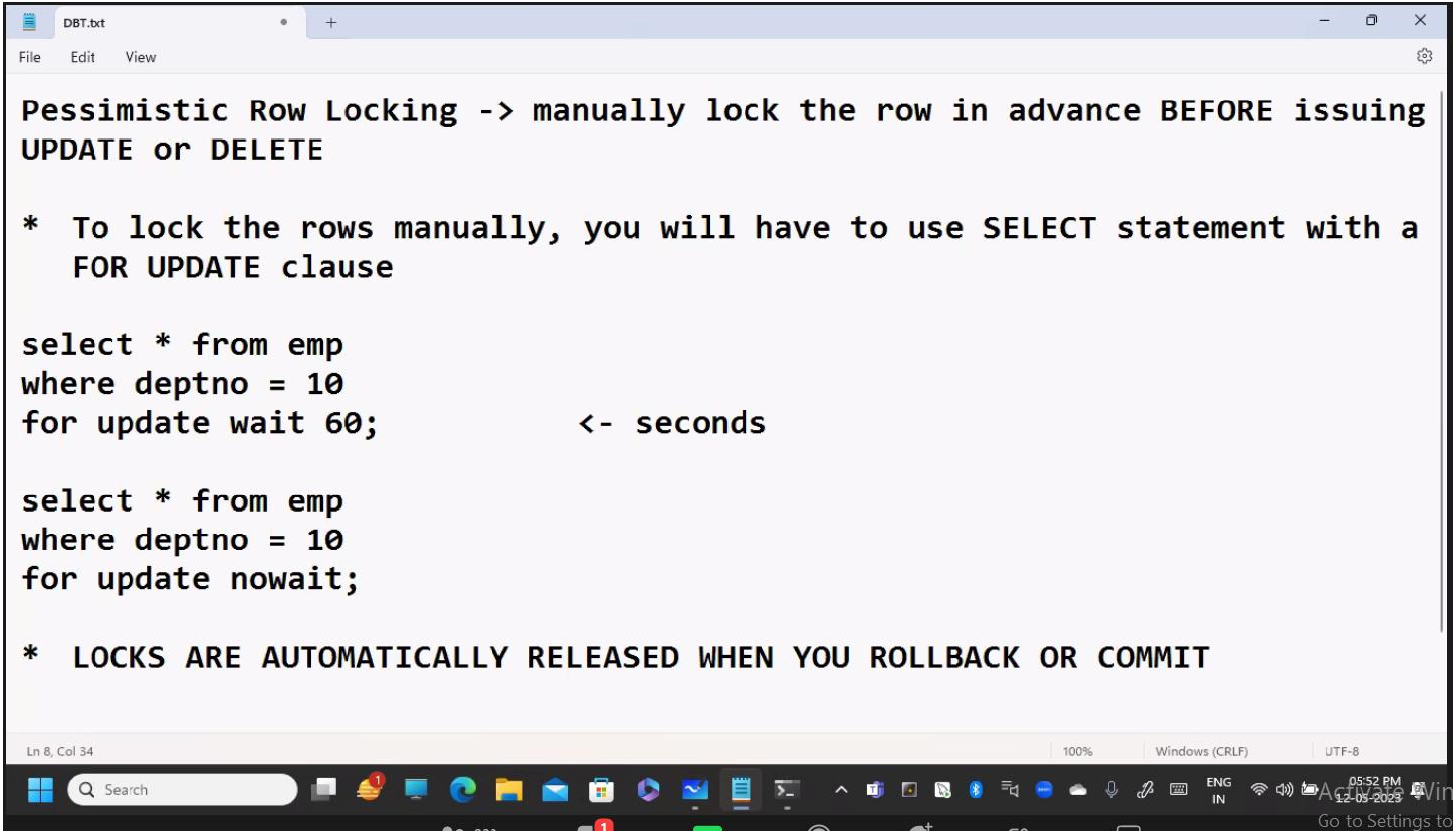
```
user1> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

user1> insert into dept values(50, 'Training', 'CDAC
');

1 row created.

user1> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
        50 Training       CDAC

user1> commit;

Commit complete.
```

```
user2> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

user2> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON

user2> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
        50 Training       CDAC
```



```
user1> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
        50 Training       CDAC

user1> update dept set dname = 'Sleeping'
  2  where deptno = 50;

1 row updated.

user1> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
        50 Sleeping       CDAC

user1> commit;

Commit complete.
```

```
user2> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
        50 Training       CDAC

user2> select * from dept;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
        50 Training       CDAC

user2> delete from dept where deptno = 50;

1 row deleted.

user2> commit;

Commit complete.
```

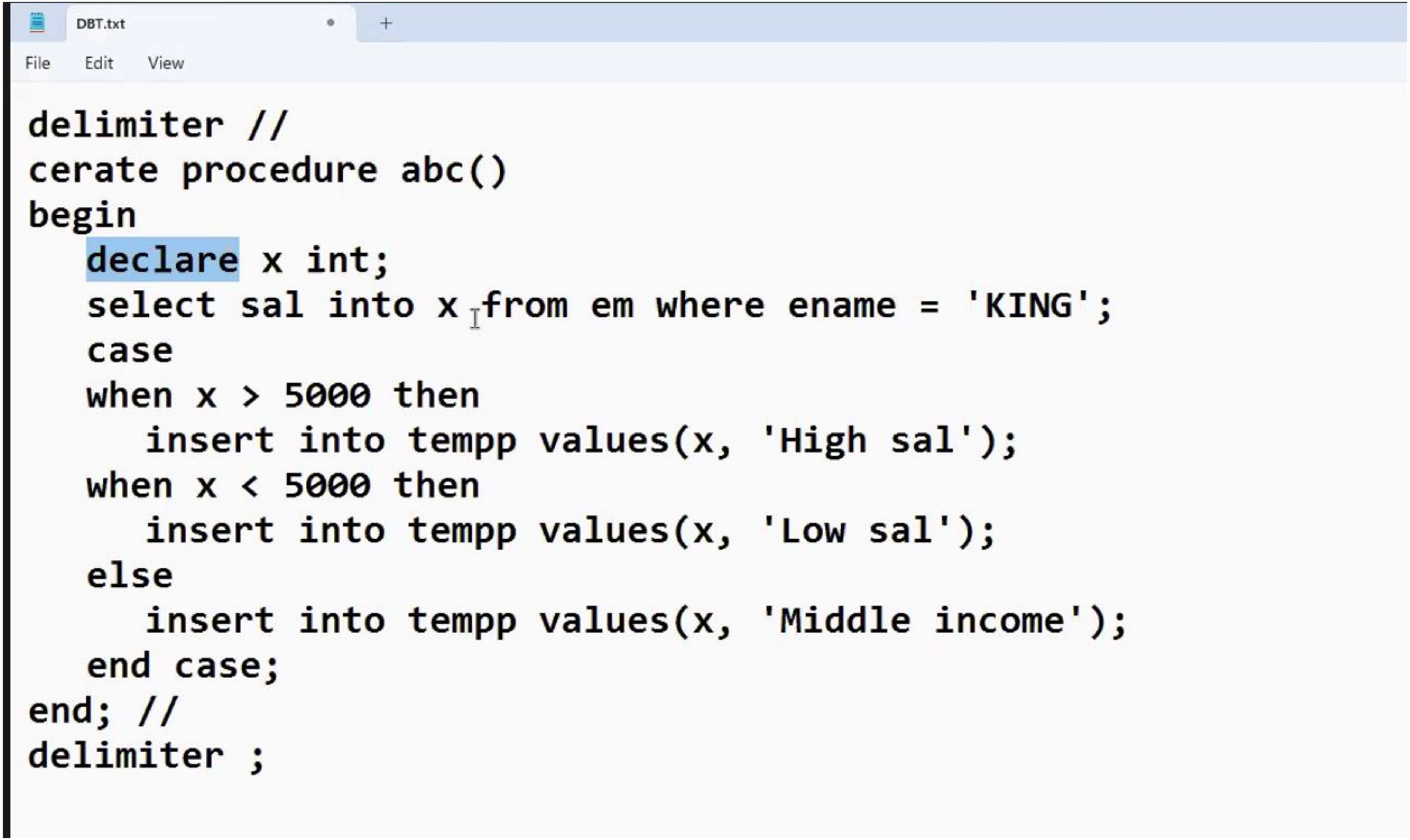


Pessimistic Row Locking -> manually lock the row in advance BEFORE issuing UPDATE or DELETE

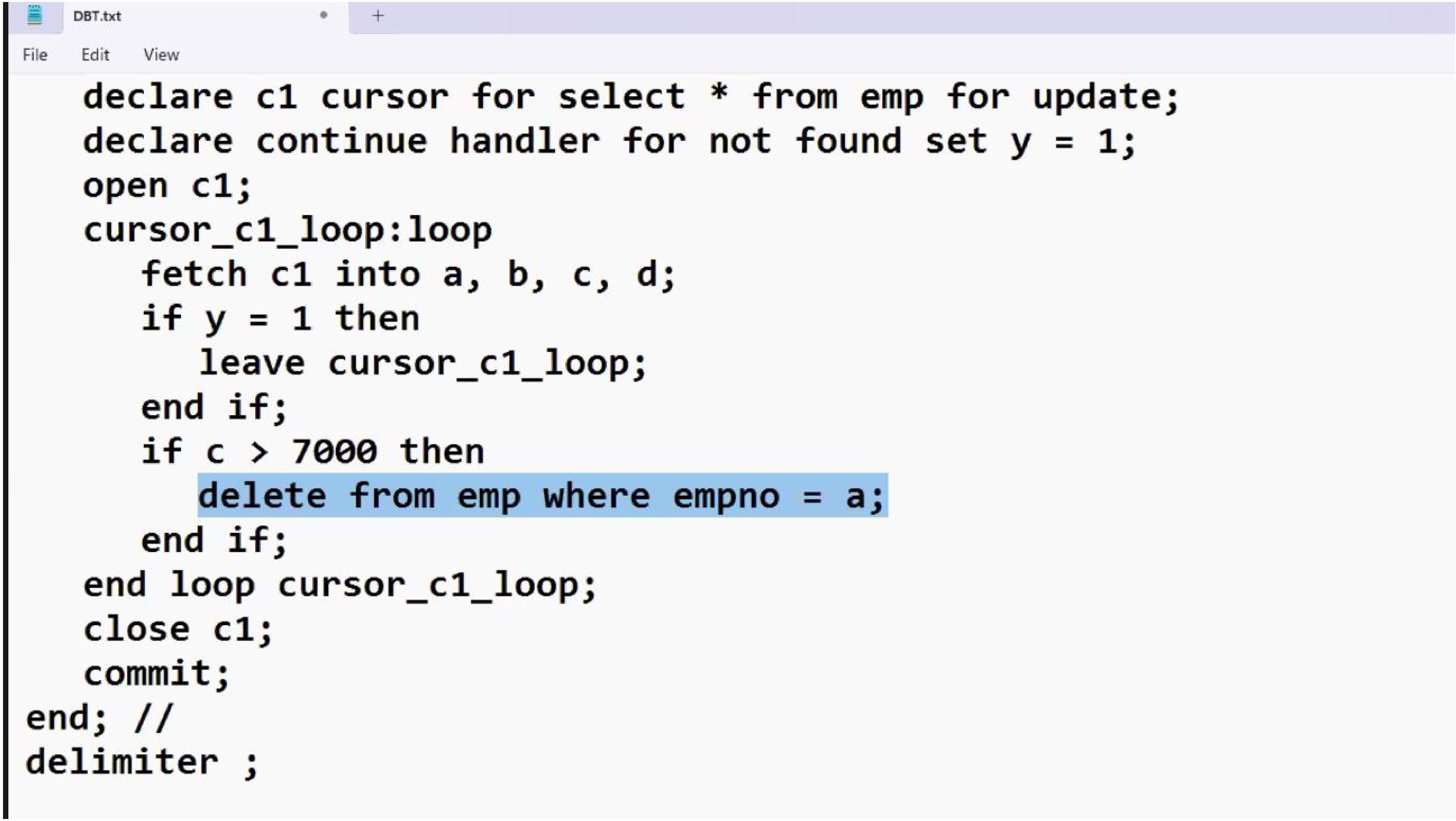
* To lock the rows manually, you will have to use SELECT statement with a FOR UPDATE clause

```
select * from emp
where deptno = 10
for update wait 60;            <- seconds

select * from emp
where deptno = 10
for update nowait;
```

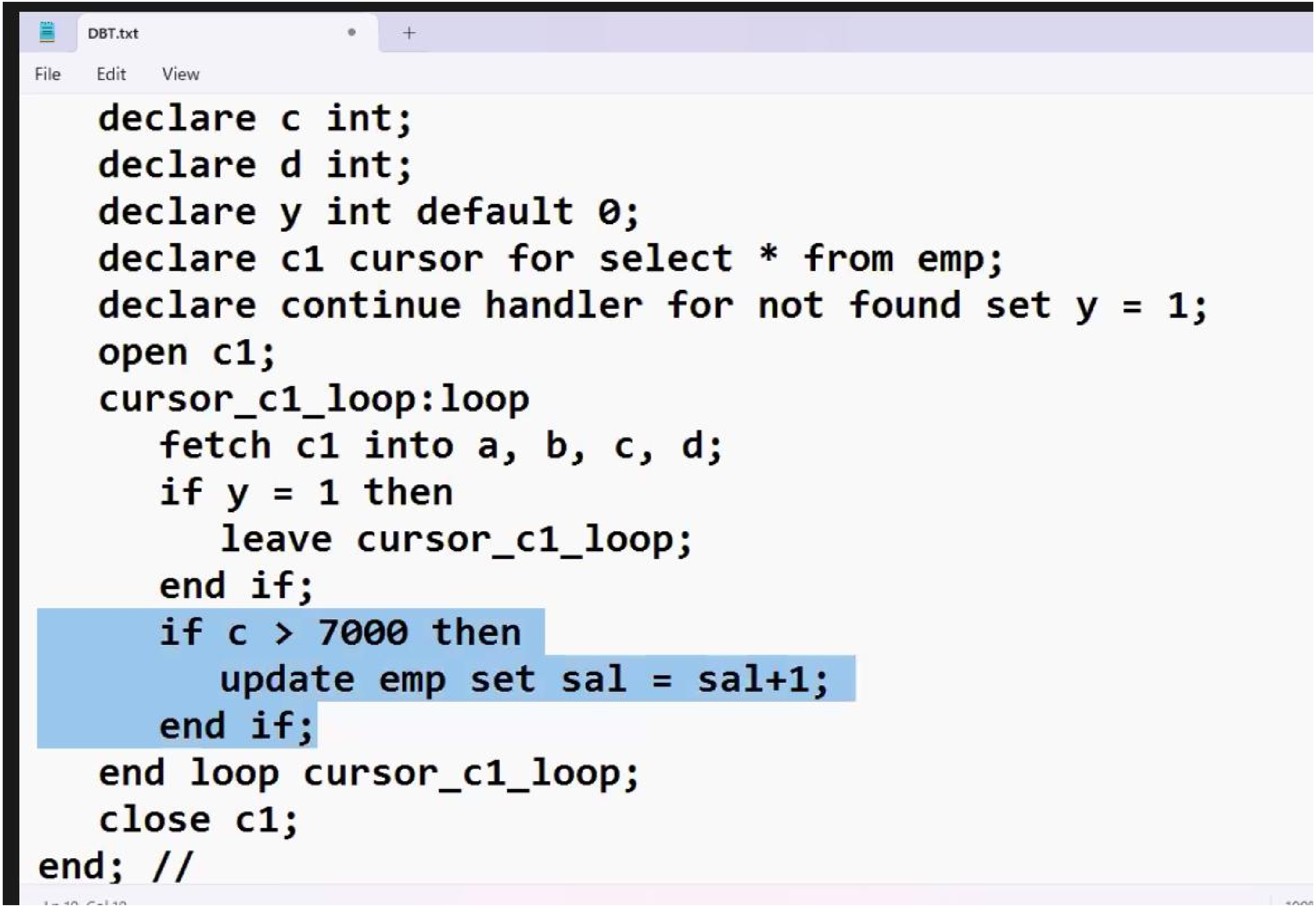
* LOCKS ARE AUTOMATICALLY RELEASED WHEN YOU ROLLBACK OR COMMIT



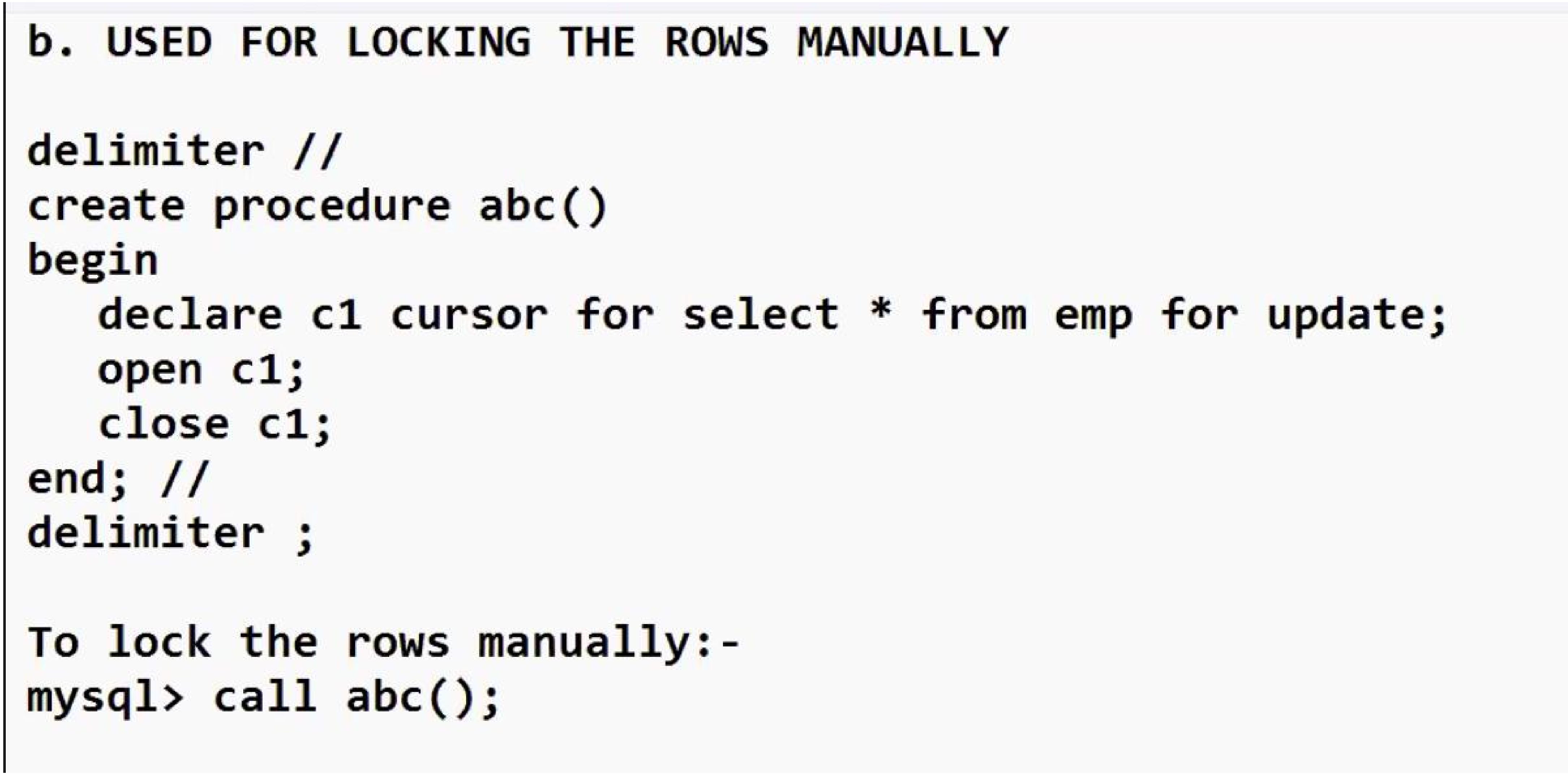
```
delimiter //
cerate procedure abc()
begin
   declare x int;
   select sal into x from em where ename = 'KING';
   case
   when x > 5000 then
      insert into tempp values(x, 'High sal');
   when x < 5000 then
      insert into tempp values(x, 'Low sal');
   else
      insert into tempp values(x, 'Middle income');
   end case;
end; //
delimiter ;
```



```
   declare c1 cursor for select * from emp for update;
   declare continue handler for not found set y = 1;
   open c1;
   cursor_c1_loop:loop
      fetch c1 into a, b, c, d;
      if y = 1 then
         leave cursor_c1_loop;
      end if;
      if c > 7000 then
         delete from emp where empno = a;
      end if;
   end loop cursor_c1_loop;
   close c1;
   commit;
end; //
delimiter ;
```



```
   declare c int;
   declare d int;
   declare y int default 0;
   declare c1 cursor for select * from emp;
   declare continue handler for not found set y = 1;
   open c1;
   cursor_c1_loop:loop
      fetch c1 into a, b, c, d;
      if y = 1 then
         leave cursor_c1_loop;
      end if;
      if c > 7000 then
         update emp set sal = sal+1;
      end if;
   end loop cursor_c1_loop;
   close c1;
end; //
```

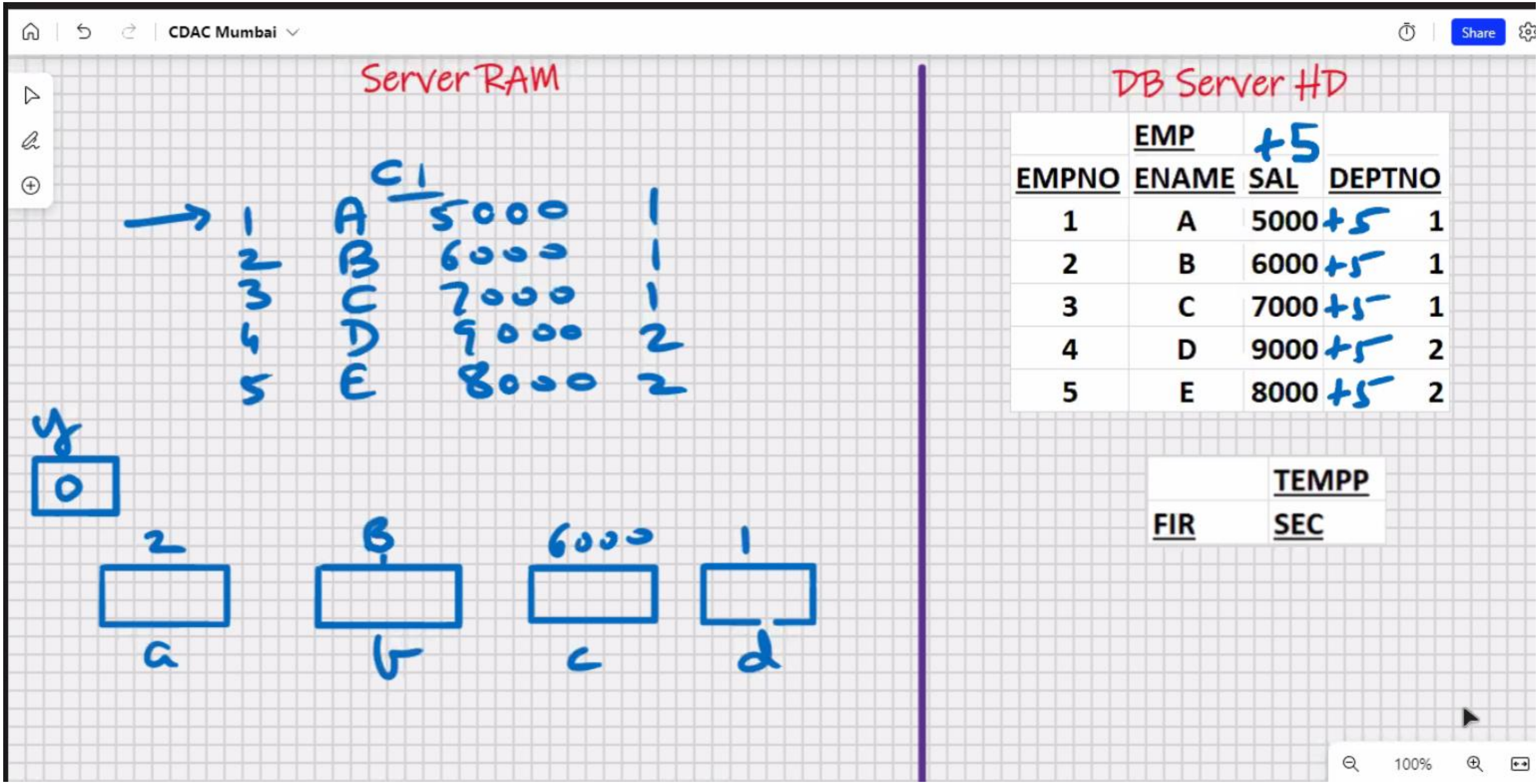


b. USED FOR LOCKING THE ROWS MANUALLY

```
delimiter //
create procedure abc()
begin
   declare c1 cursor for select * from emp for update;
   open c1;
   close c1;
end; //
delimiter ;
```

To lock the rows manually:-

```
mysql> call abc();
```



Server RAM

C1

1	A	5000	1
2	B	6000	1
3	C	7000	1
4	D	9000	2
5	E	8000	2

y

0

a	b	c	d
2	B	6000	1

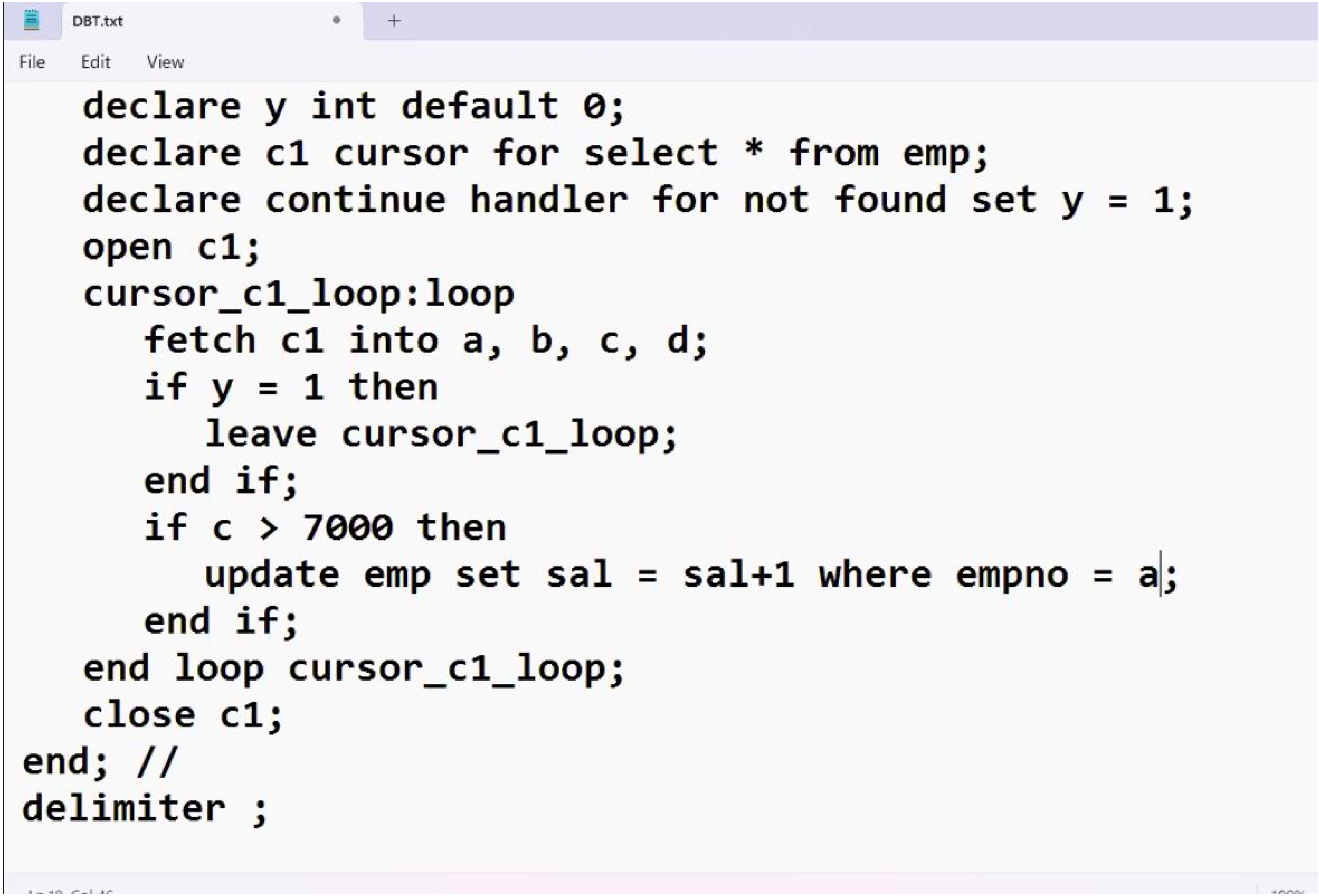
DB Server HD

EMP +5

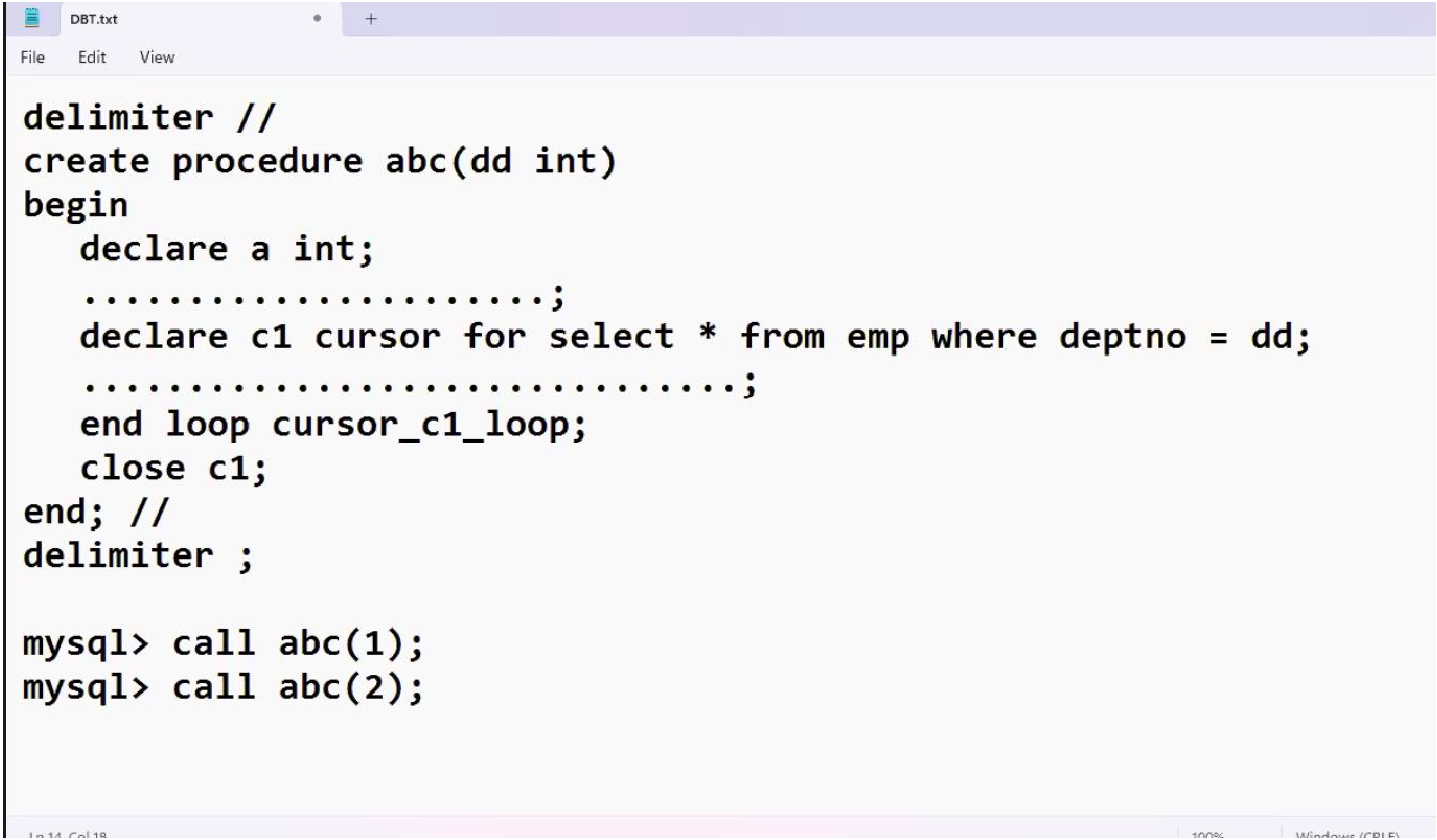
EMPNO	**ENAME**	**SAL**	**DEPTNO**
1	A	5000+5	1
2	B	6000+5	1
3	C	7000+5	1
4	D	9000+5	2
5	E	8000+5	2

TEMPP

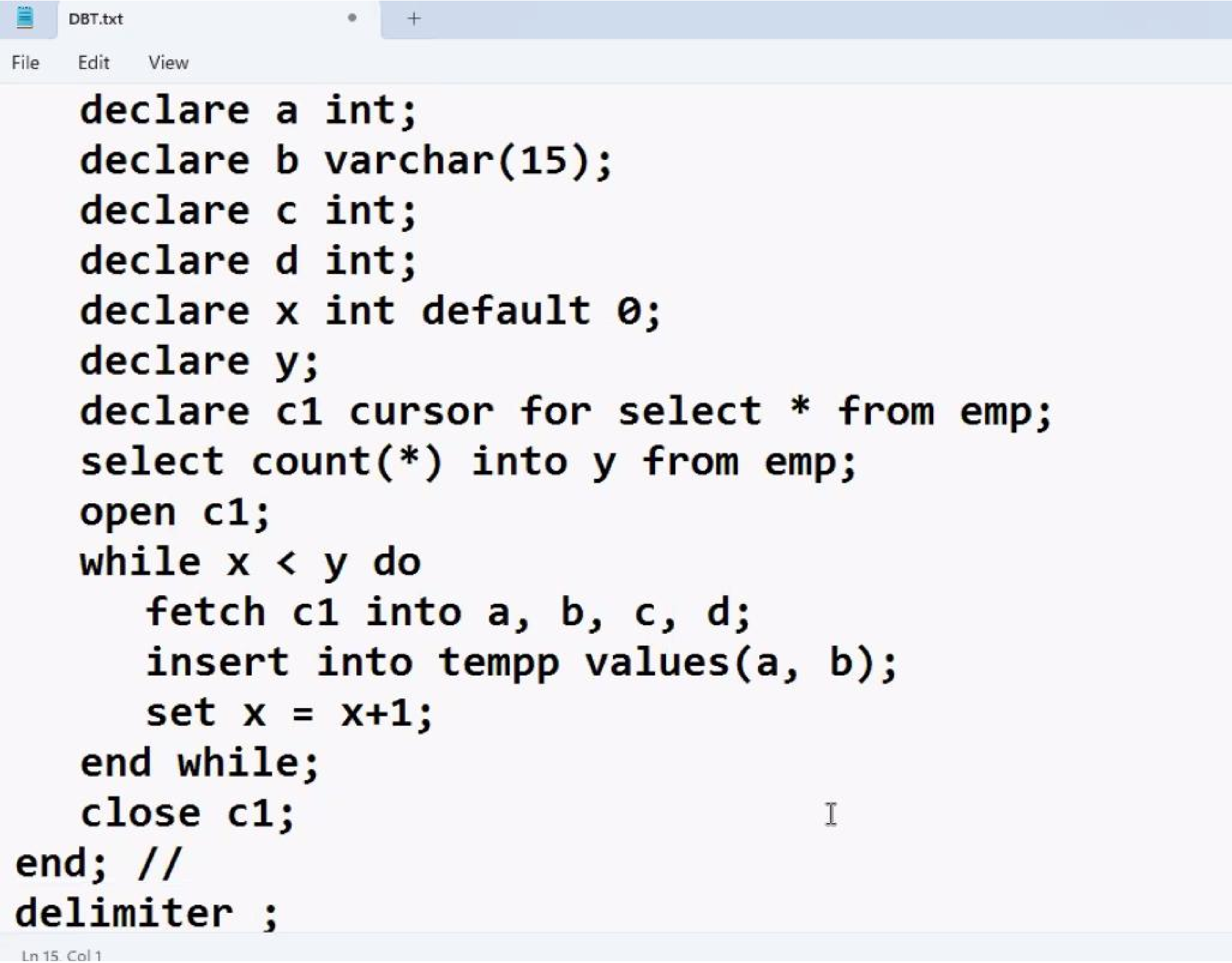
FIR	**SEC**



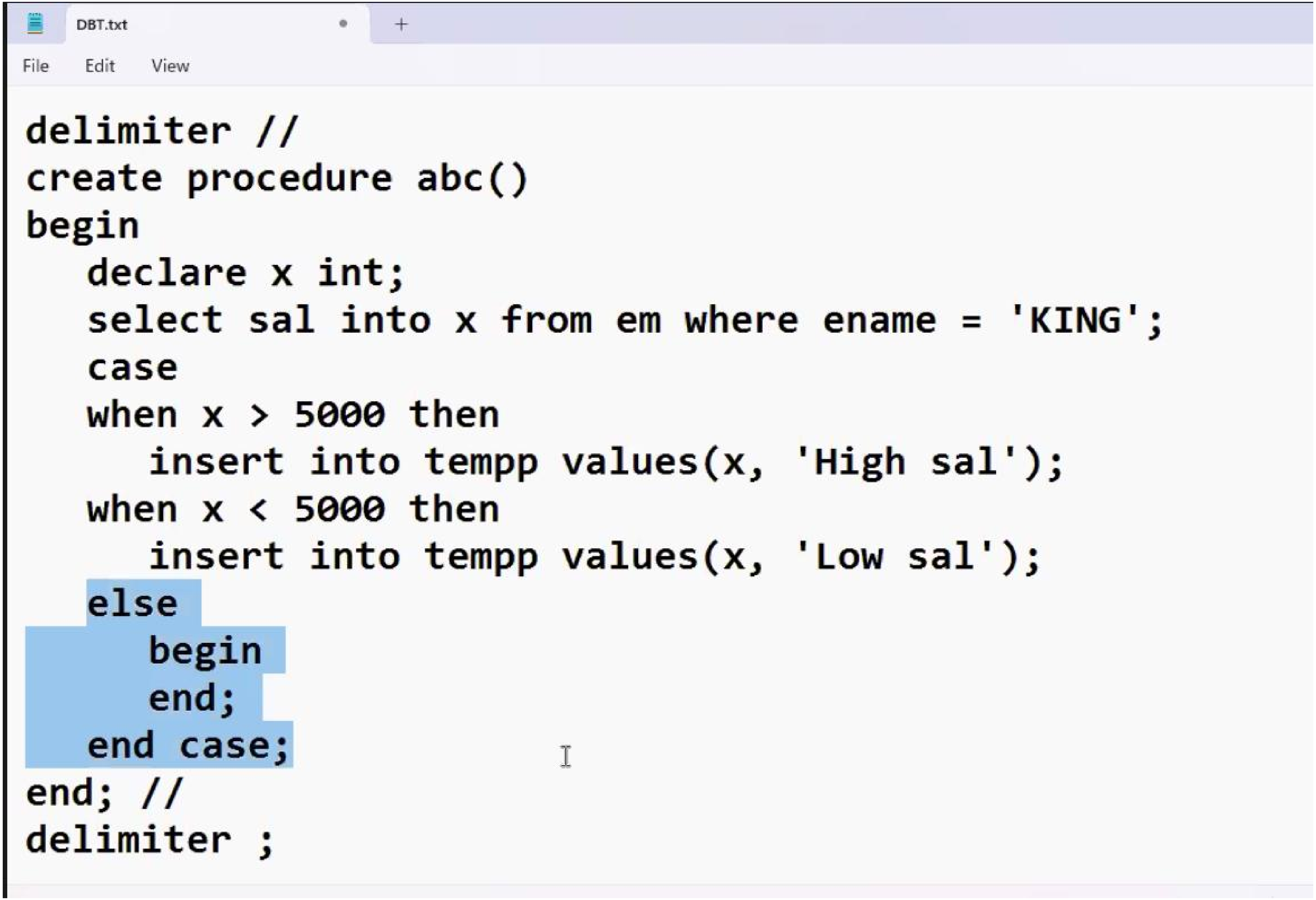
```
   declare y int default 0;
   declare c1 cursor for select * from emp;
   declare continue handler for not found set y = 1;
   open c1;
   cursor_c1_loop:loop
      fetch c1 into a, b, c, d;
      if y = 1 then
         leave cursor_c1_loop;
      end if;
      if c > 7000 then
         update emp set sal = sal+1 where empno = a;
      end if;
   end loop cursor_c1_loop;
   close c1;
end; //
delimiter ;
```



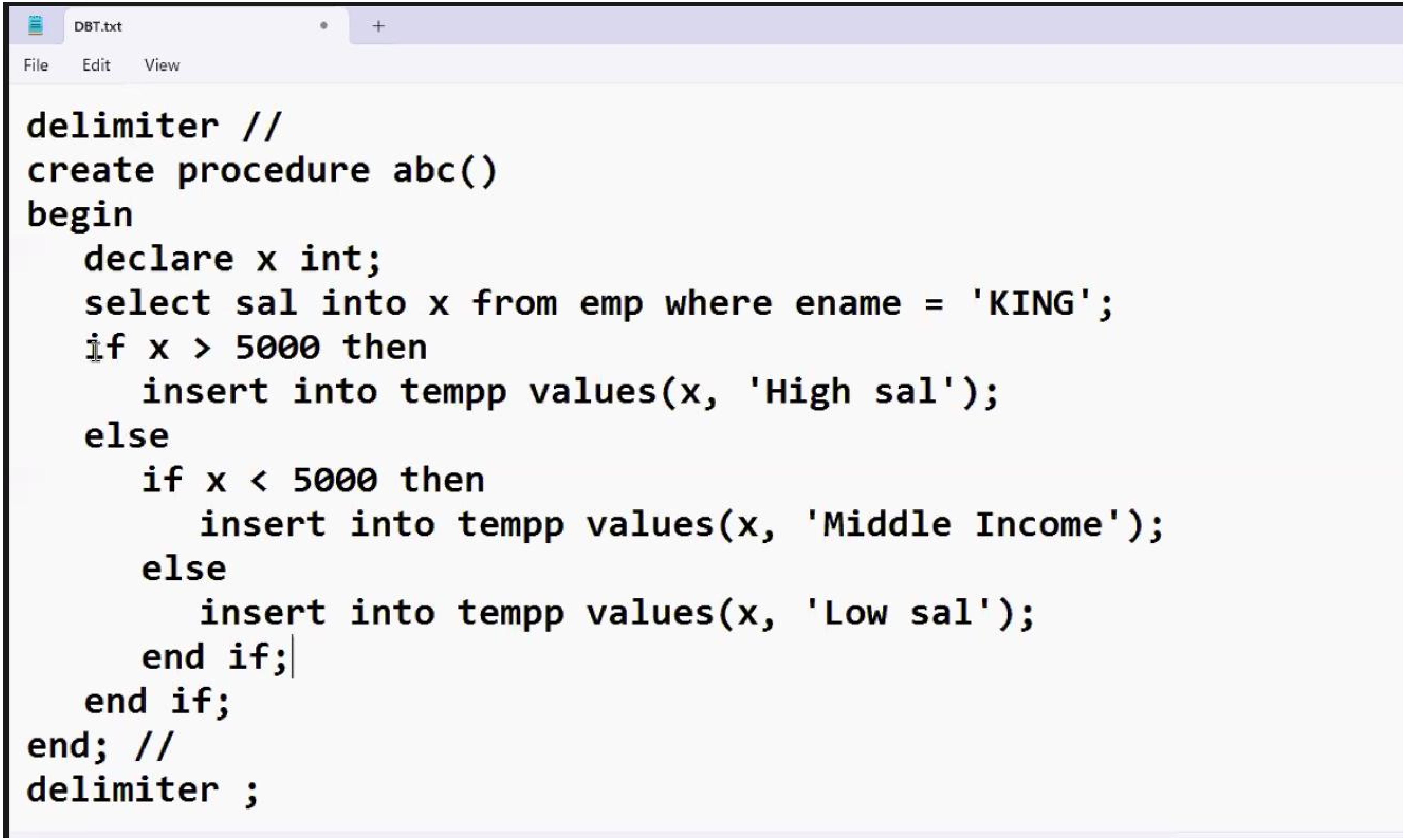
```
delimiter //
create procedure abc(dd int)
begin
   declare a int;
   .......................;
   declare c1 cursor for select * from emp where deptno = dd;
   ...............................;
   end loop cursor_c1_loop;
   close c1;
end; //
delimiter ;

mysql> call abc(1);
mysql> call abc(2);
```



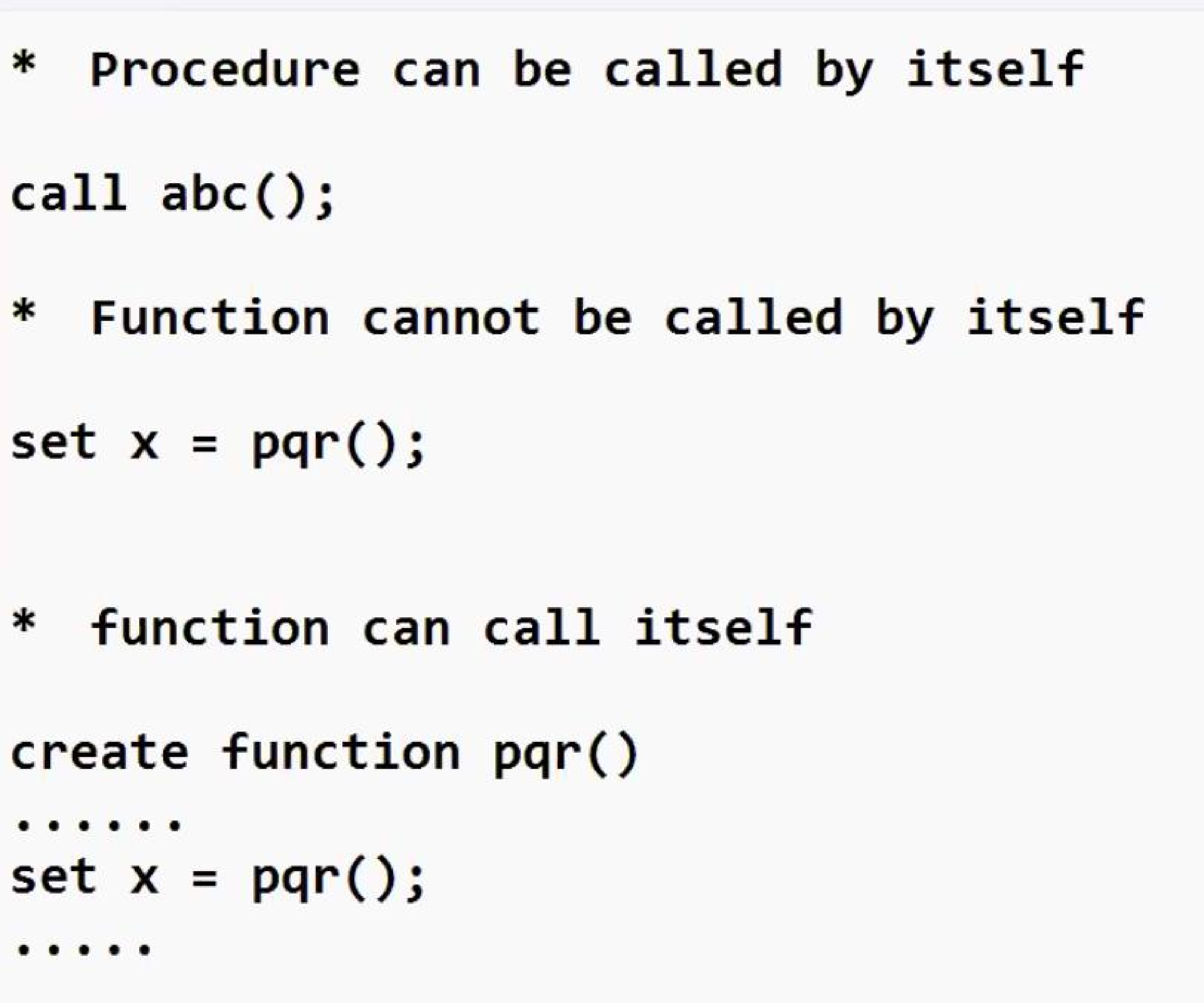
```
   declare a int;
   declare b varchar(15);
   declare c int;
   declare d int;
   declare x int default 0;
   declare y;
   declare c1 cursor for select * from emp;
   select count(*) into y from emp;
   open c1;
   while x < y do
      fetch c1 into a, b, c, d;
      insert into tempp values(a, b);
      set x = x+1;
   end while;
   close c1;
end; //
delimiter ;
```



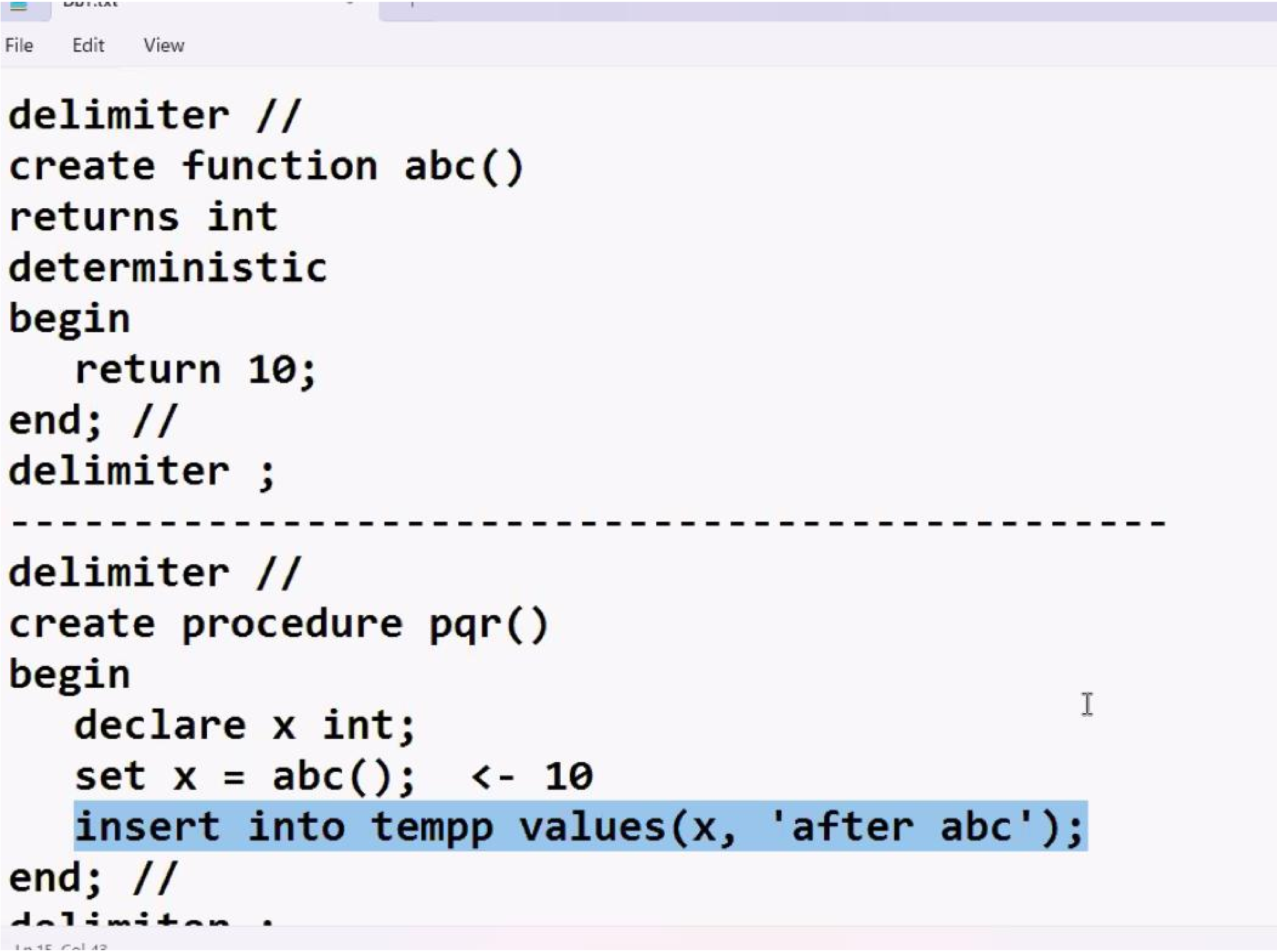
```
delimiter //
create procedure abc()
begin
   declare x int;
   select sal into x from em where ename = 'KING';
   case
   when x > 5000 then
      insert into tempp values(x, 'High sal');
   when x < 5000 then
      insert into tempp values(x, 'Low sal');
   else
      begin
      end;
   end case;
end; //
delimiter ;
```



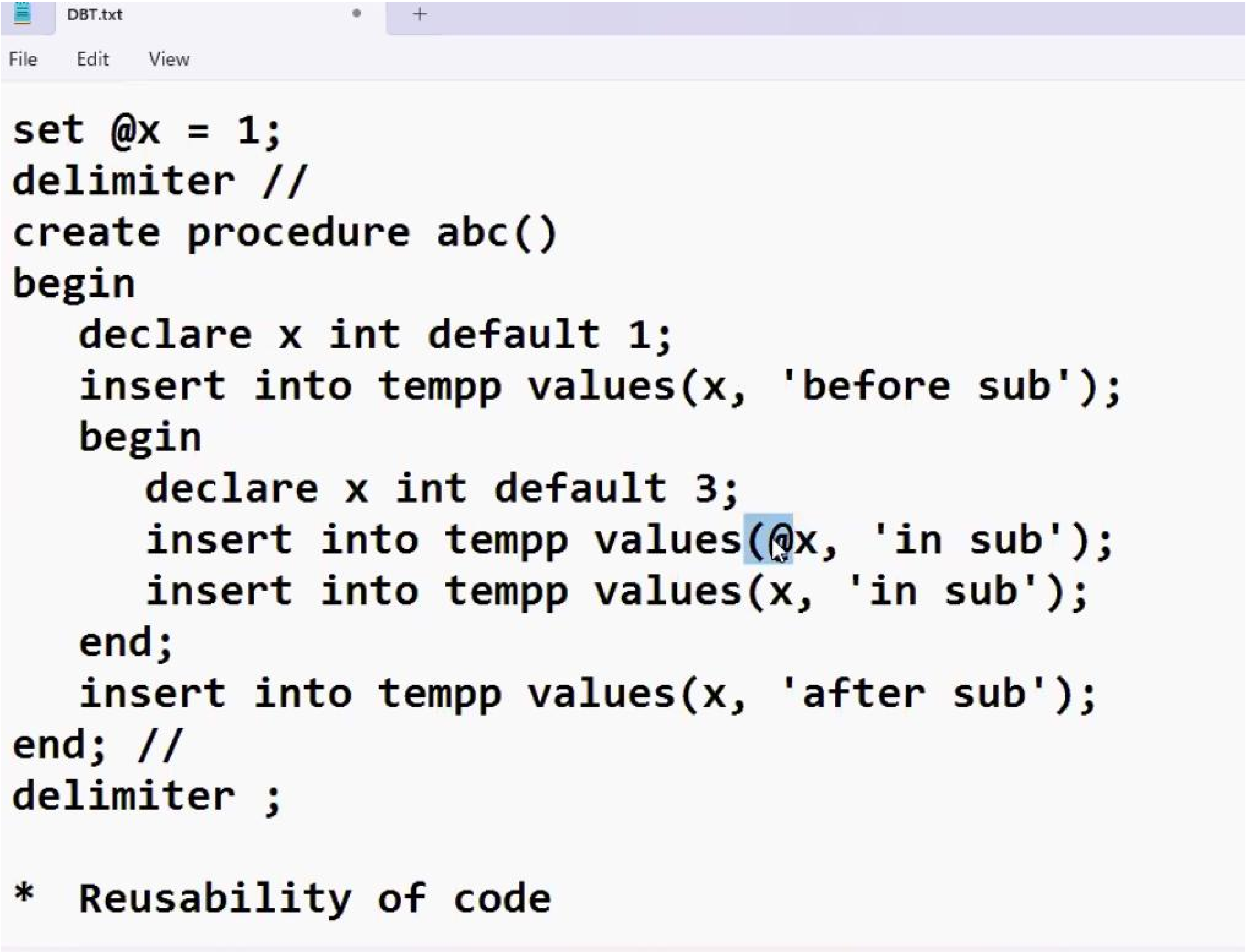
```
delimiter //
create procedure abc()
begin
   declare x int;
   select sal into x from emp where ename = 'KING';
   if x > 5000 then
      insert into tempp values(x, 'High sal');
   else
      if x < 5000 then
         insert into tempp values(x, 'Middle Income');
      else
         insert into tempp values(x, 'Low sal');
      end if;
   end if;
end; //
delimiter ;
```



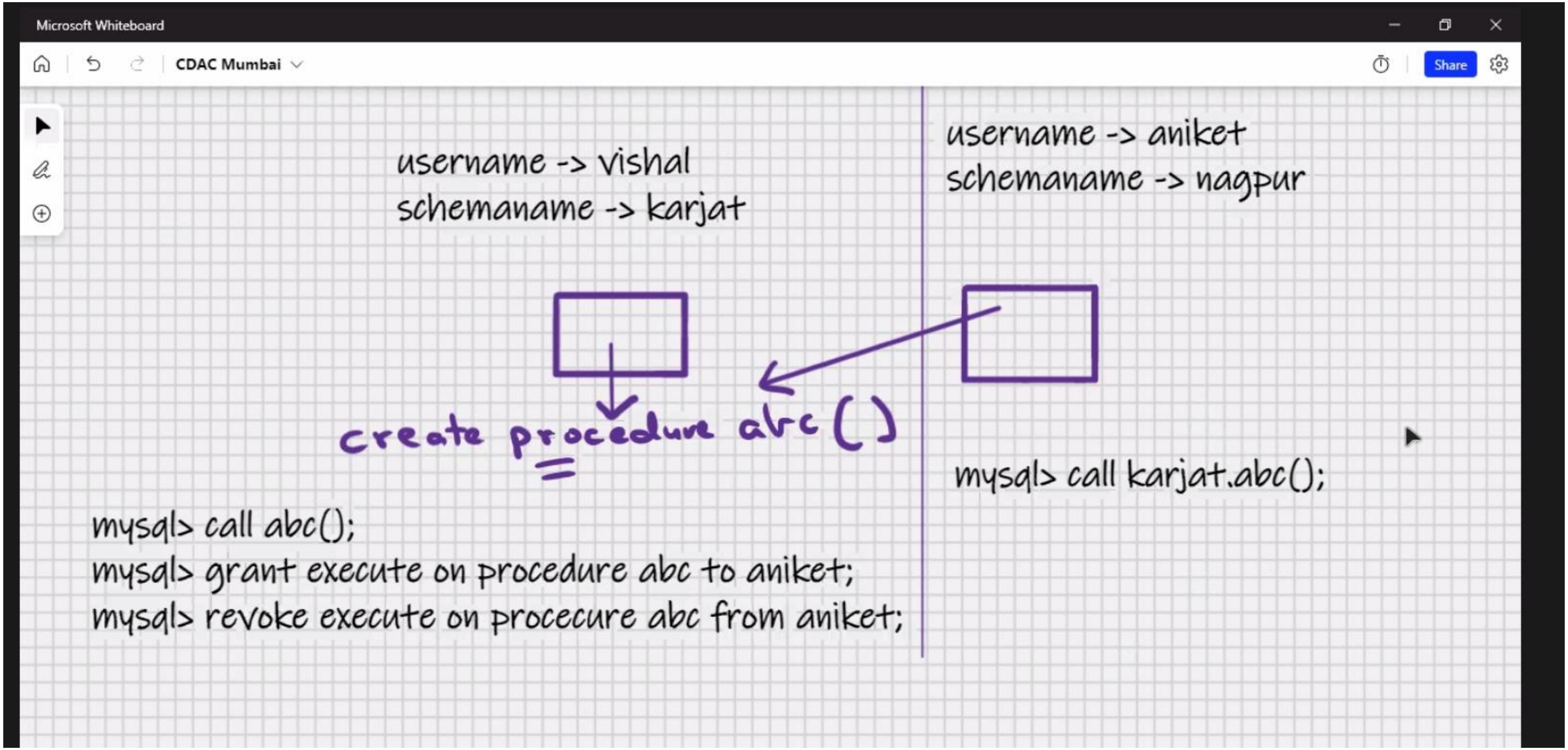
```
*  Procedure can be called by itself

call abc();

*  Function cannot be called by itself

set x = pqr();


*  function can call itself

create function pqr()
......
set x = pqr();
.....
```



```
delimiter //
create function abc()
returns int
deterministic
begin
   return 10;
end; //
delimiter ;
-----------------------------------------------
delimiter //
create procedure pqr()
begin
   declare x int;
   set x = abc();  <- 10
   insert into tempp values(x, 'after abc');
end; //
delimiter ;
```



```
set @x = 1;
delimiter //
create procedure abc()
begin
   declare x int default 1;
   insert into tempp values(x, 'before sub');
   begin
      declare x int default 3;
      insert into tempp values(@x, 'in sub');
      insert into tempp values(x, 'in sub');
   end;
   insert into tempp values(x, 'after sub');
end; //
delimiter ;

*  Reusability of code
```



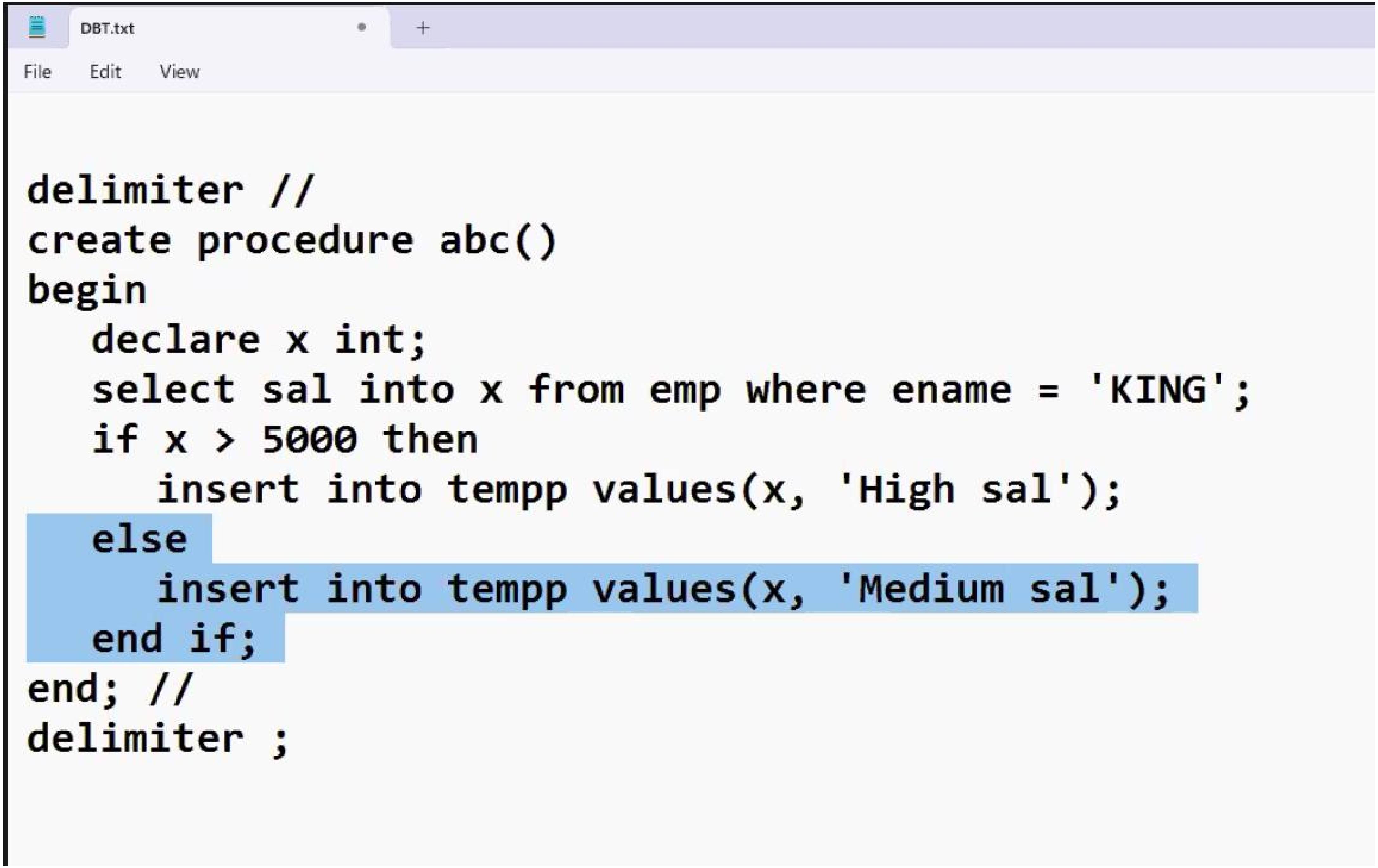
username -> vishal
schemaname -> karjat

create procedure abc()

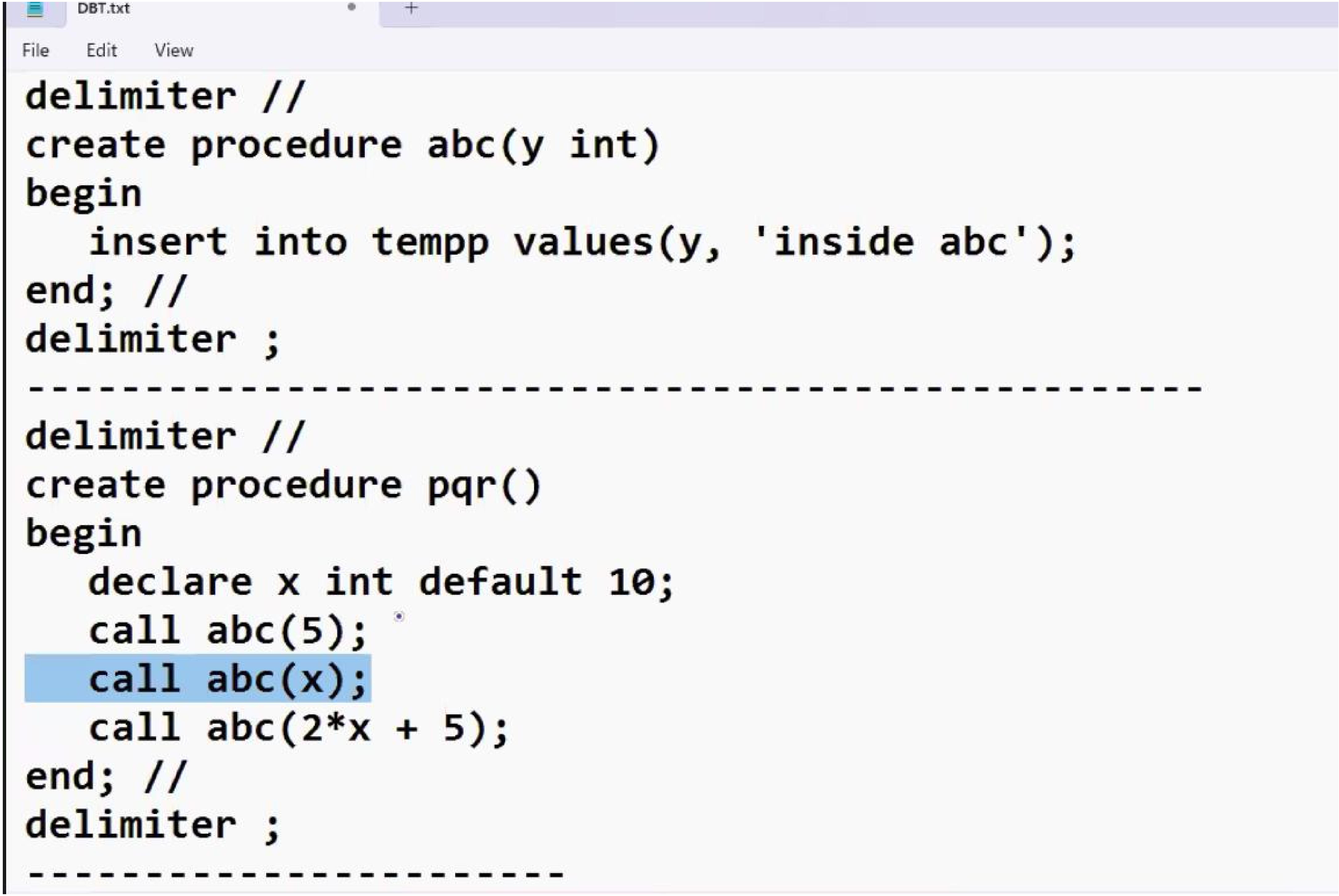
```
mysql> call abc();
mysql> grant execute on procedure abc to aniket;
mysql> revoke execute on procecure abc from aniket;
```

username -> aniket
schemaname -> nagpur

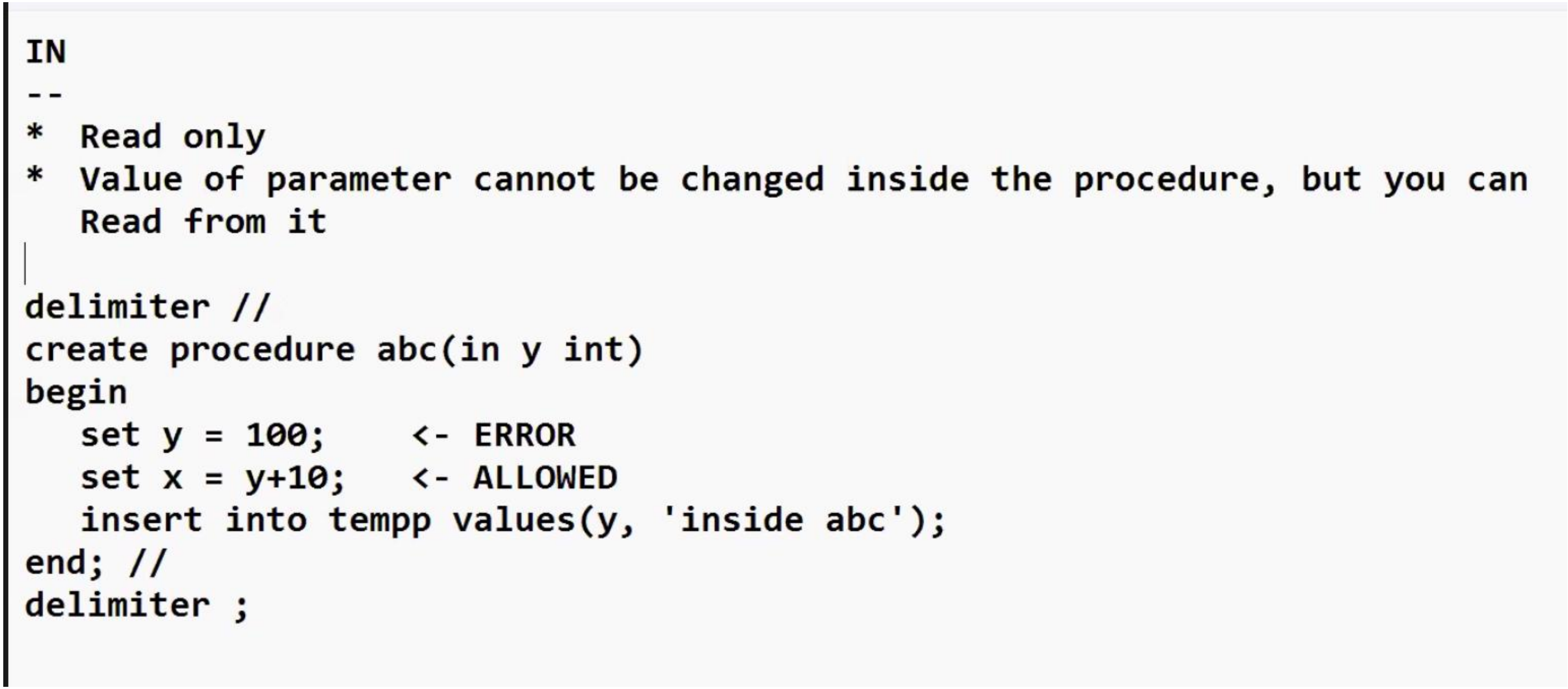
```
mysql> call karjat.abc();
```



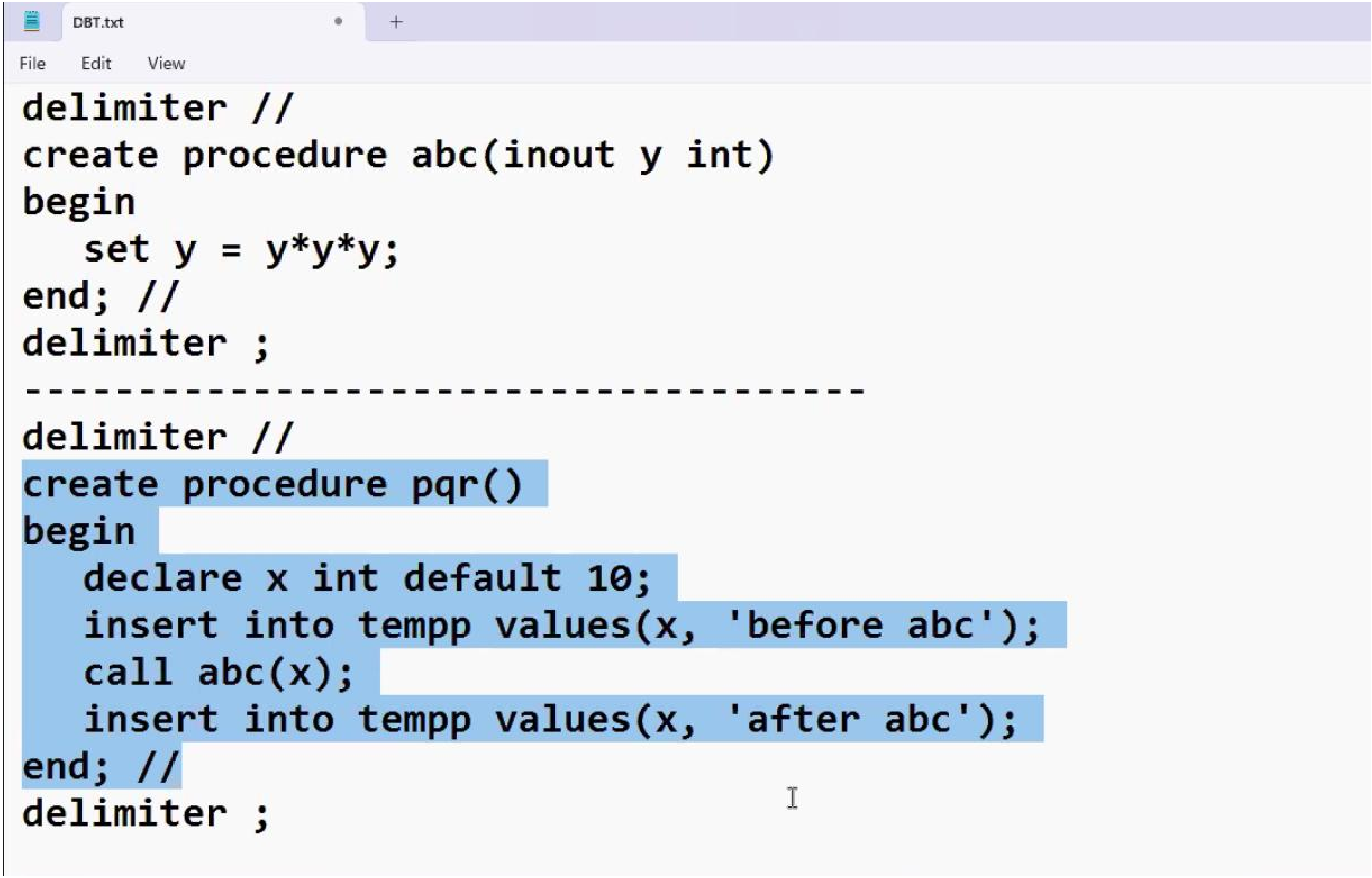
```
delimiter //
create procedure abc()
begin
   declare x int;
   select sal into x from emp where ename = 'KING';
   if x > 5000 then
      insert into tempp values(x, 'High sal');
   else
      insert into tempp values(x, 'Medium sal');
   end if;
end; //
delimiter ;
```



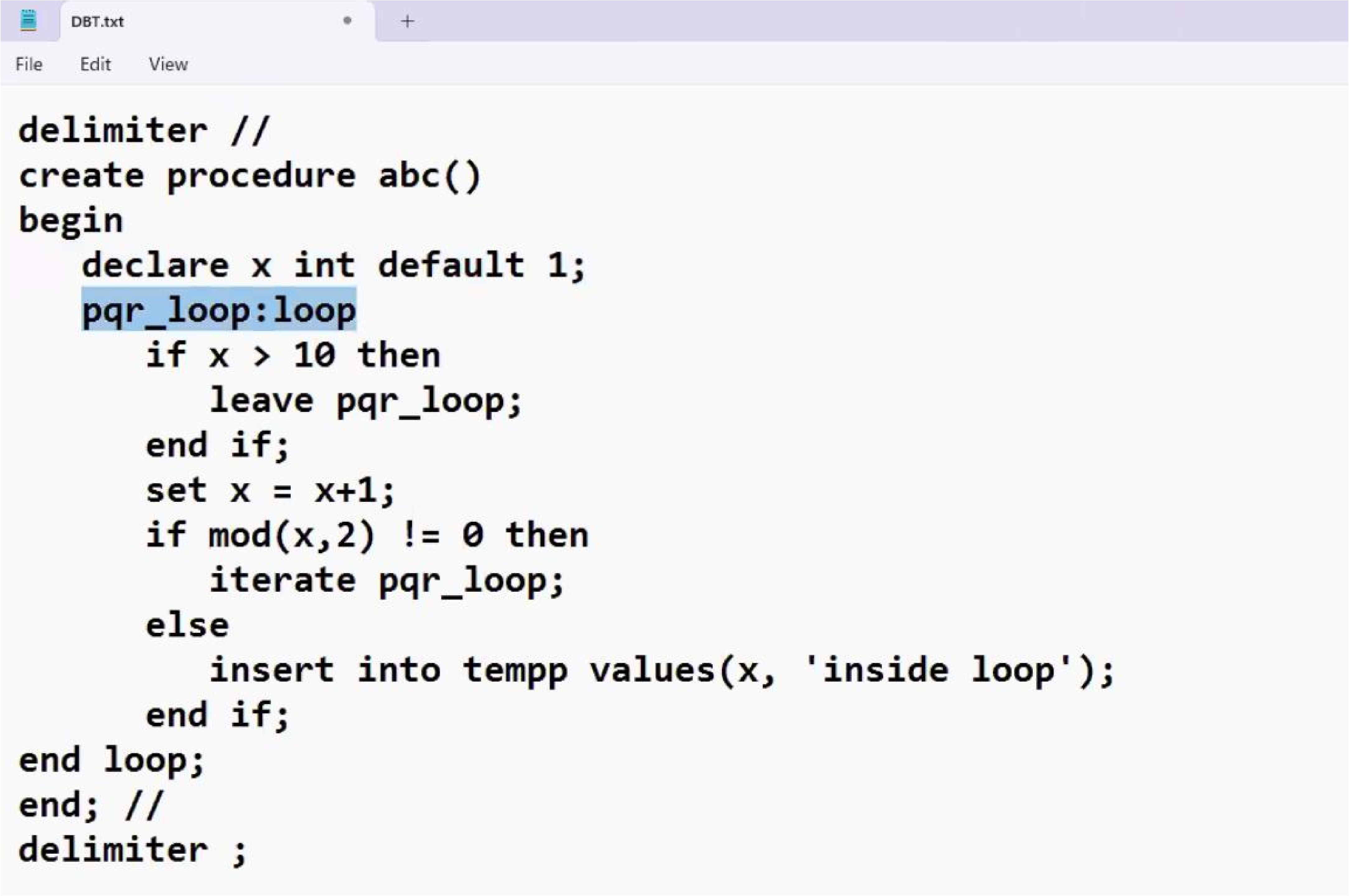
```
delimiter //
create procedure abc(y int)
begin
   insert into tempp values(y, 'inside abc');
end; //
delimiter ;
------------------------------------------------
delimiter //
create procedure pqr()
begin
   declare x int default 10;
   call abc(5);
   call abc(x);
   call abc(2*x + 5);
end; //
delimiter ;
----------------------
```



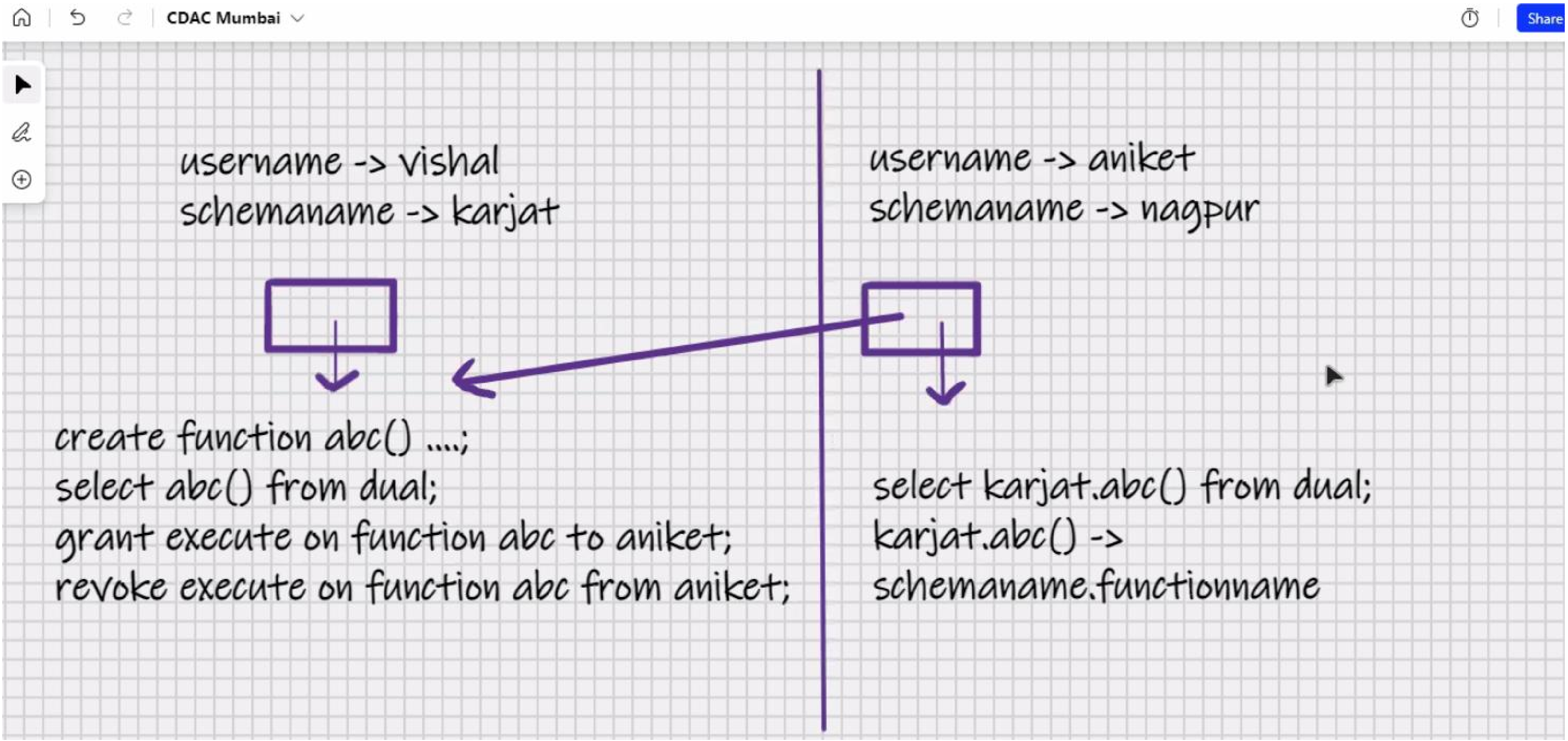
```
IN
--
*  Read only
*  Value of parameter cannot be changed inside the procedure, but you can
   Read from it

delimiter //
create procedure abc(in y int)
begin
   set y = 100;      <- ERROR
   set x = y+10;     <- ALLOWED
   insert into tempp values(y, 'inside abc');
end; //
delimiter ;
```



```
delimiter //
create procedure abc(inout y int)
begin
   set y = y*y*y;
end; //
delimiter ;
-------------------------------------
delimiter //
create procedure pqr()
begin
   declare x int default 10;
   insert into tempp values(x, 'before abc');
   call abc(x);
   insert into tempp values(x, 'after abc');
end; //
delimiter ;
```



```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   pqr_loop:loop
      if x > 10 then
         leave pqr_loop;
      end if;
      set x = x+1;
      if mod(x,2) != 0 then
         iterate pqr_loop;
      else
         insert into tempp values(x, 'inside loop');
      end if;
end loop;
end; //
delimiter ;
```

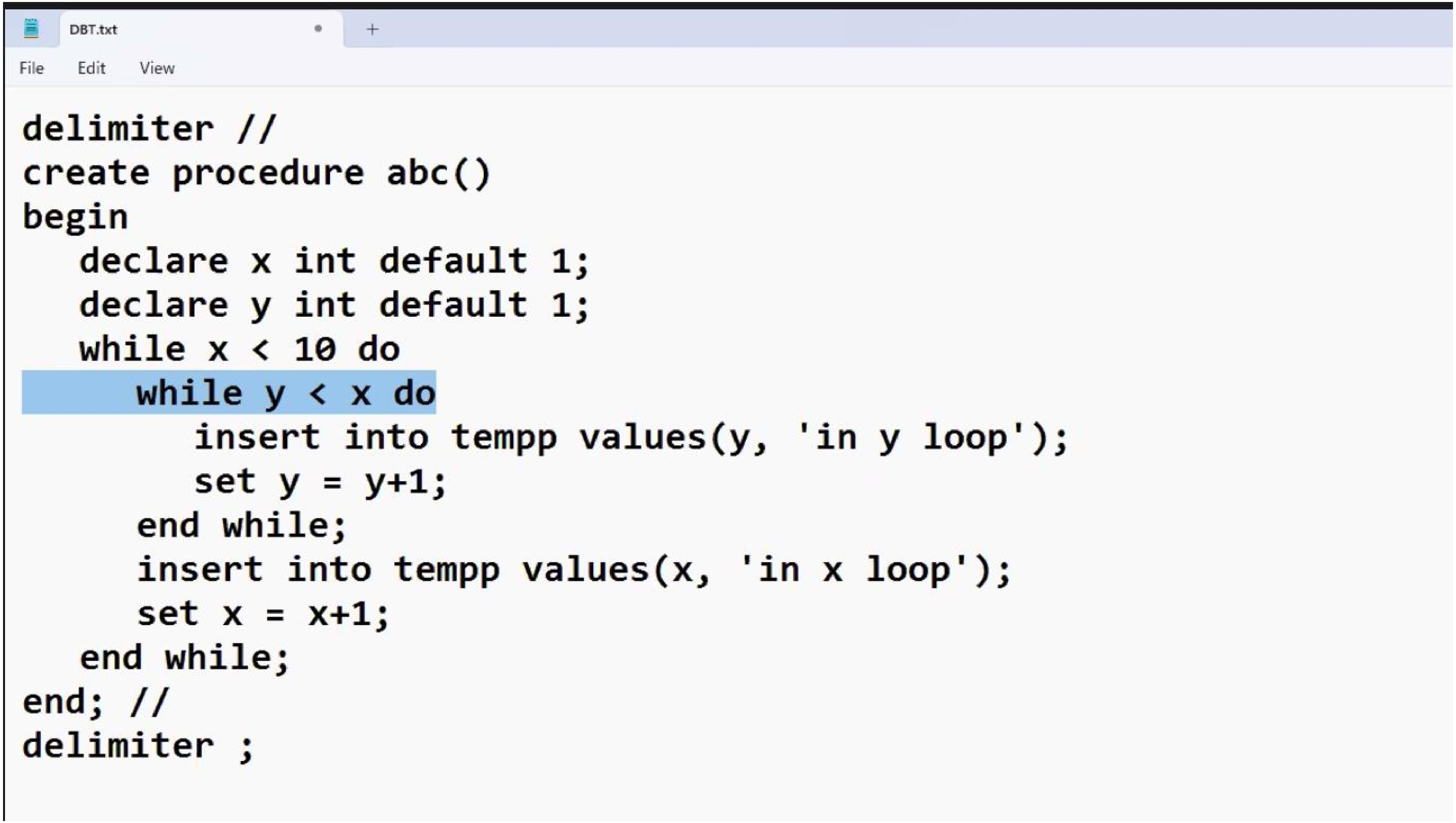


username -> vishal
schemaname -> karjat

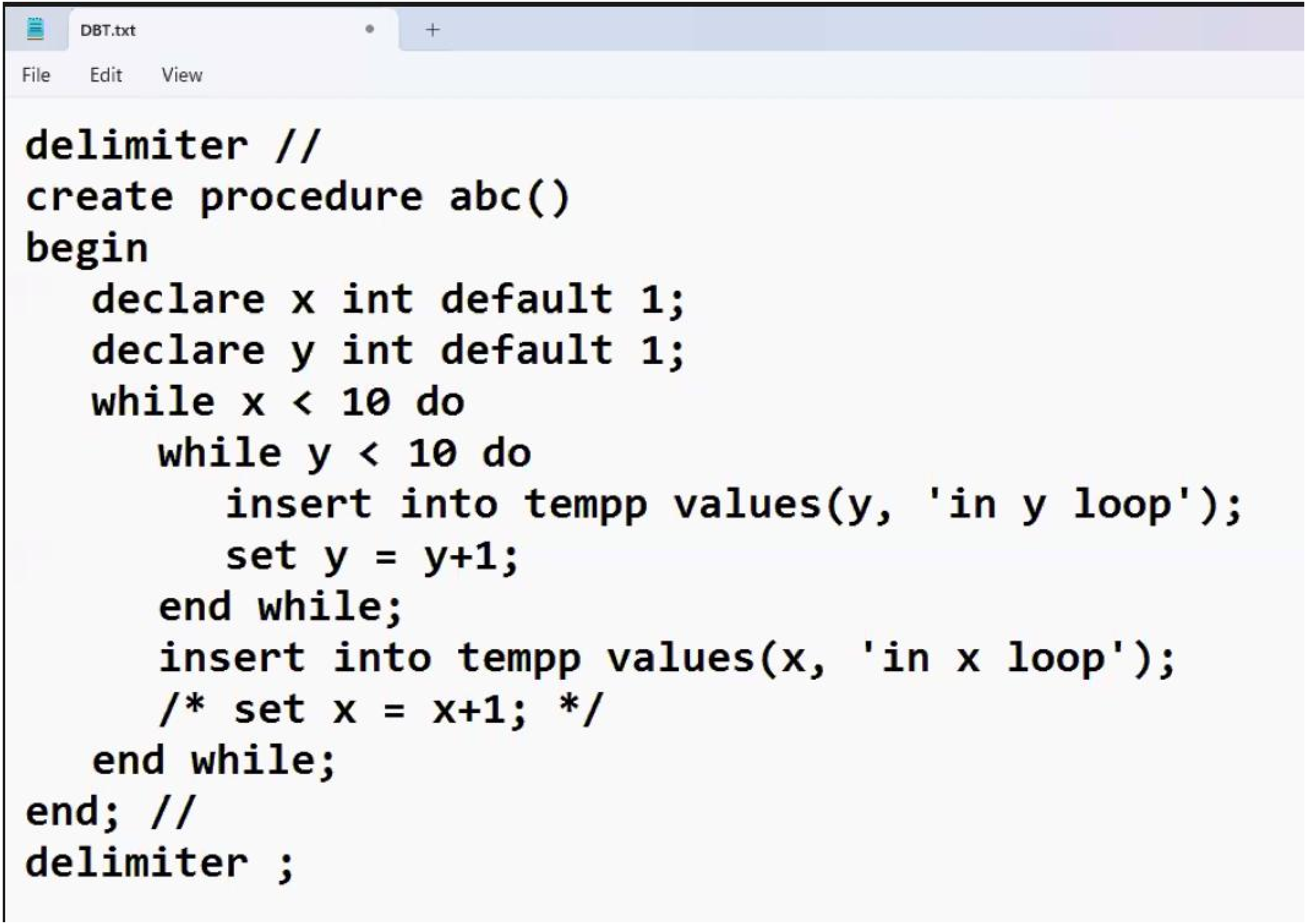
create function abc() ....;
select abc() from dual;
grant execute on function abc to aniket;
revoke execute on function abc from aniket;

username -> aniket
schemaname -> nagpur

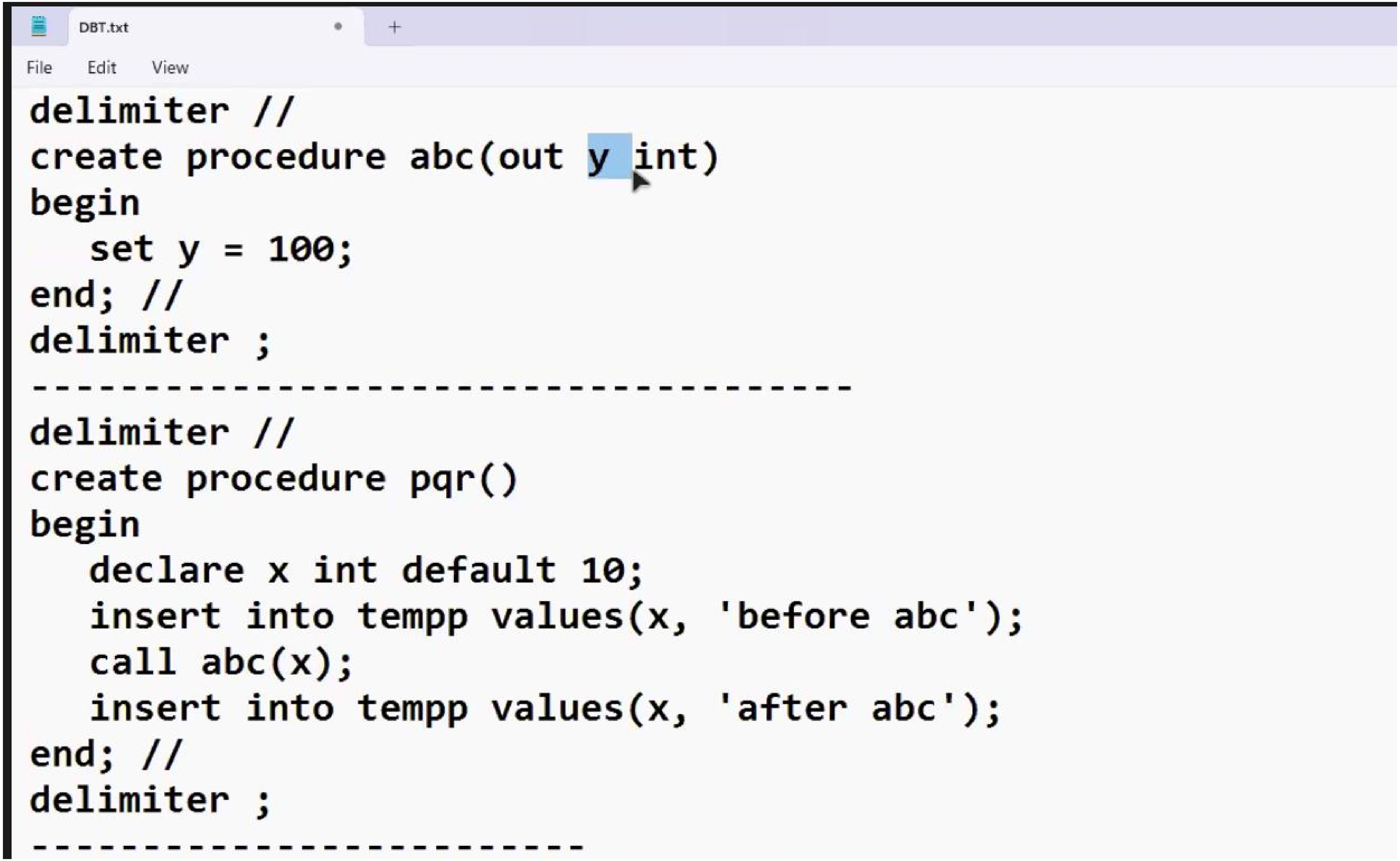
select karjat.abc() from dual;
karjat.abc() ->
schemaname.functionname



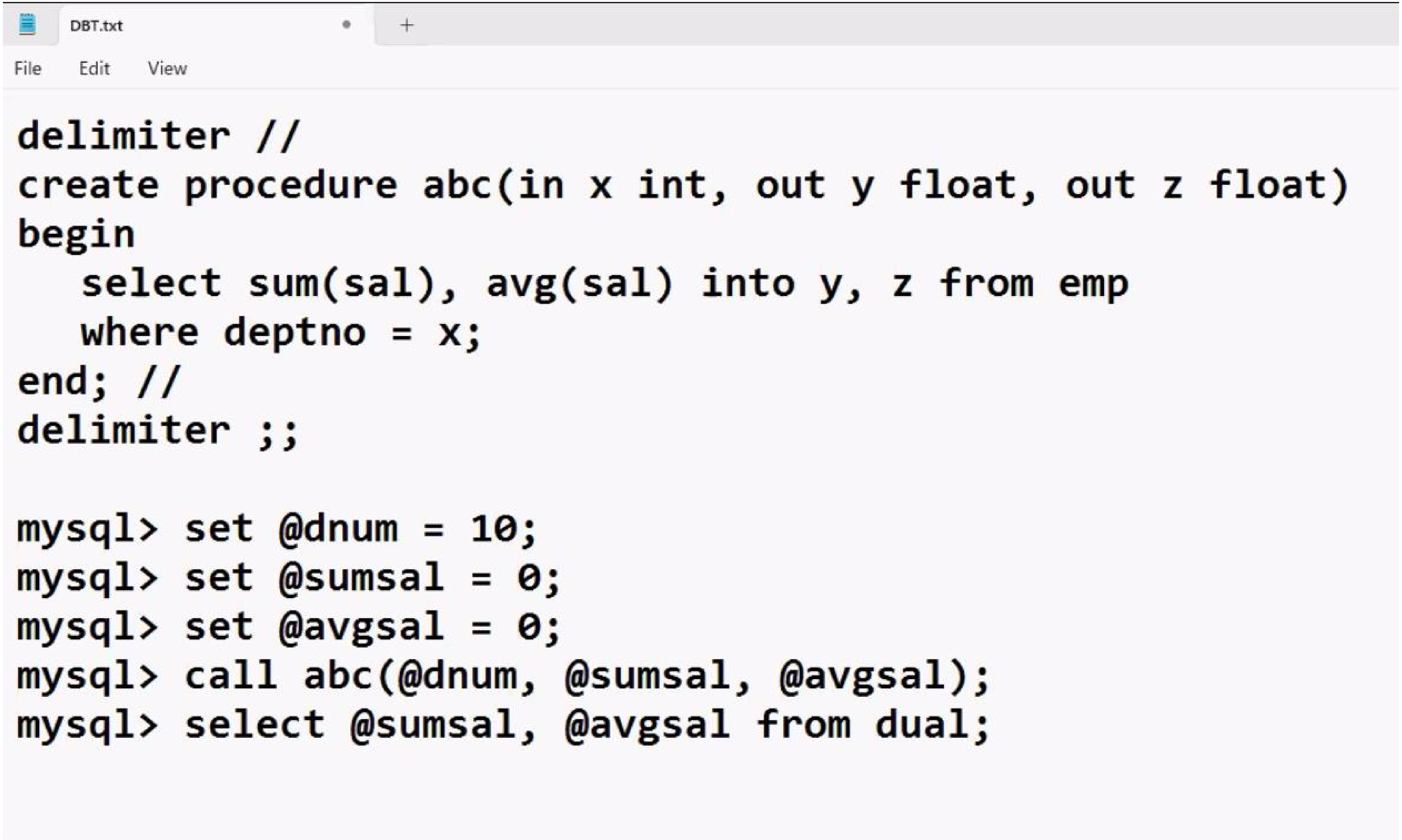
```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   declare y int default 1;
   while x < 10 do
      while y < x do
         insert into tempp values(y, 'in y loop');
         set y = y+1;
      end while;
      insert into tempp values(x, 'in x loop');
      set x = x+1;
   end while;
end; //
delimiter ;
```



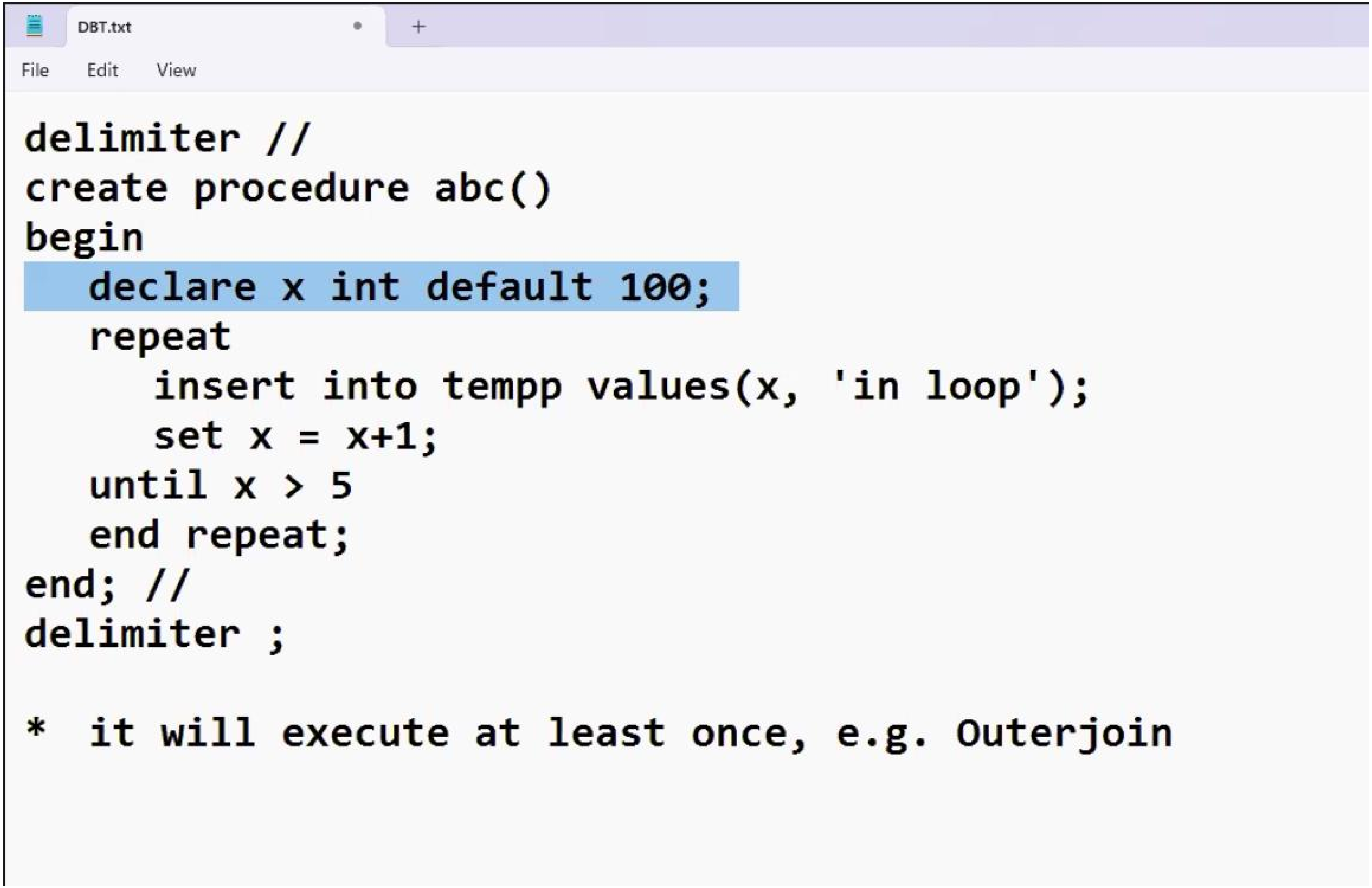
```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   declare y int default 1;
   while x < 10 do
      while y < 10 do
         insert into tempp values(y, 'in y loop');
         set y = y+1;
      end while;
      insert into tempp values(x, 'in x loop');
      /* set x = x+1; */
   end while;
end; //
delimiter ;
```



```
delimiter //
create procedure abc(out y int)
begin
   set y = 100;
end; //
delimiter ;
-----------------------------------
delimiter //
create procedure pqr()
begin
   declare x int default 10;
   insert into tempp values(x, 'before abc');
   call abc(x);
   insert into tempp values(x, 'after abc');
end; //
delimiter ;
-----------------------
```

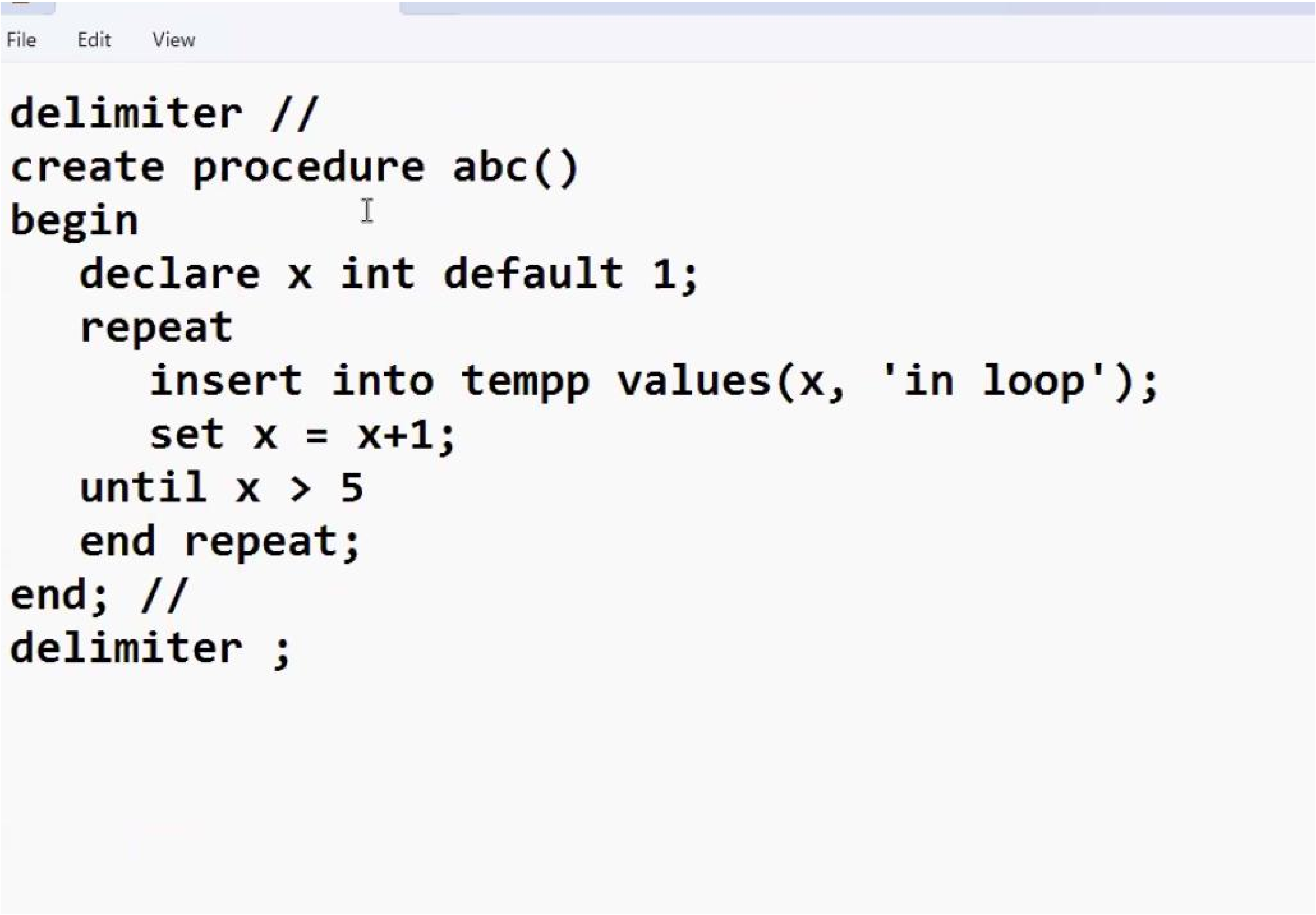


```
delimiter //
create procedure abc(in x int, out y float, out z float)
begin
  select sum(sal), avg(sal) into y, z from emp
  where deptno = x;
end; //
delimiter ;;

mysql> set @dnum = 10;
mysql> set @sumsal = 0;
mysql> set @avgsal = 0;
mysql> call abc(@dnum, @sumsal, @avgsal);
mysql> select @sumsal, @avgsal from dual;
```

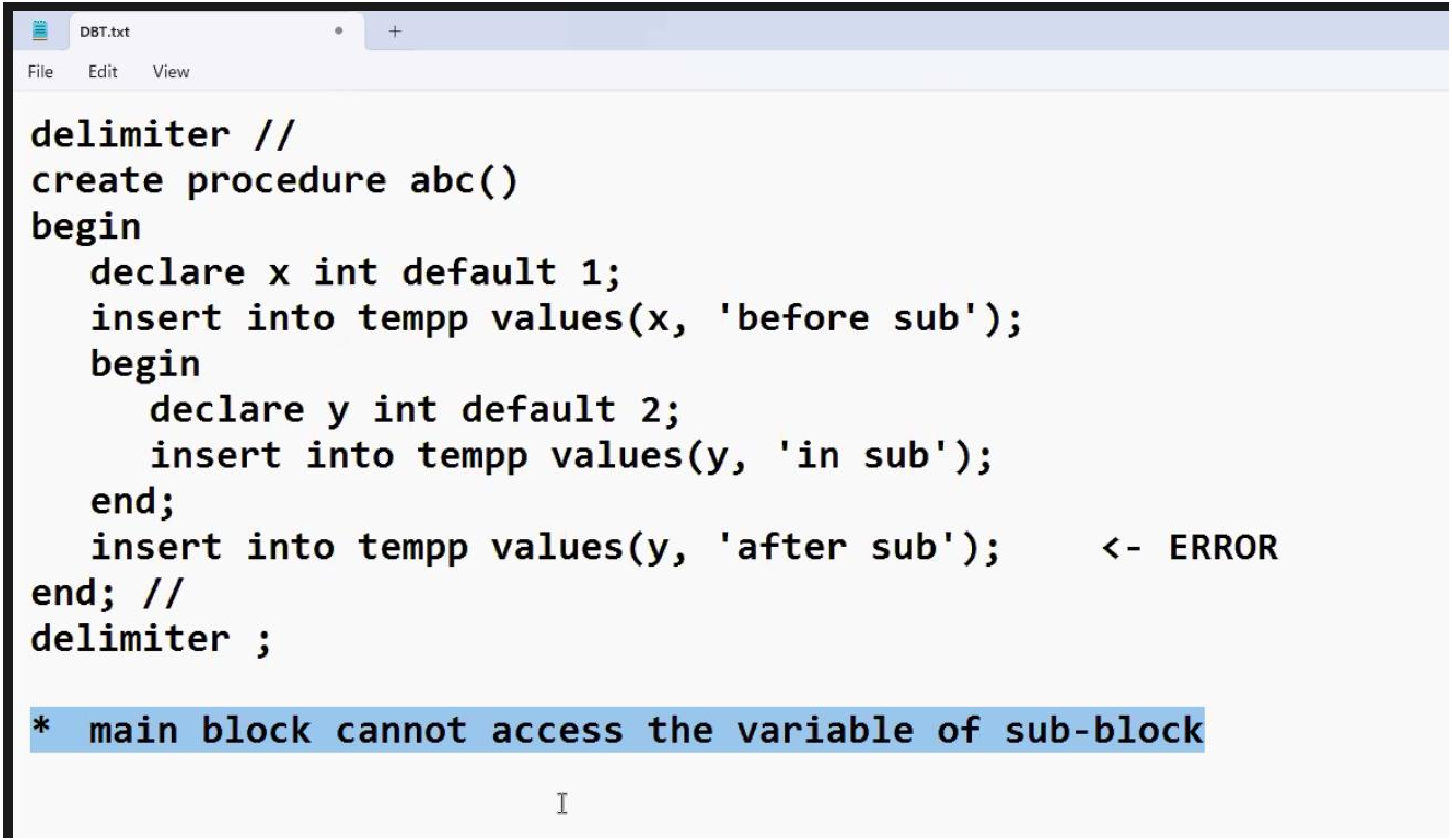


```
delimiter //
create procedure abc()
begin
   declare x int default 100;
   repeat
      insert into tempp values(x, 'in loop');
      set x = x+1;
   until x > 5
   end repeat;
end; //
delimiter ;
```

* it will execute at least once, e.g. Outerjoin

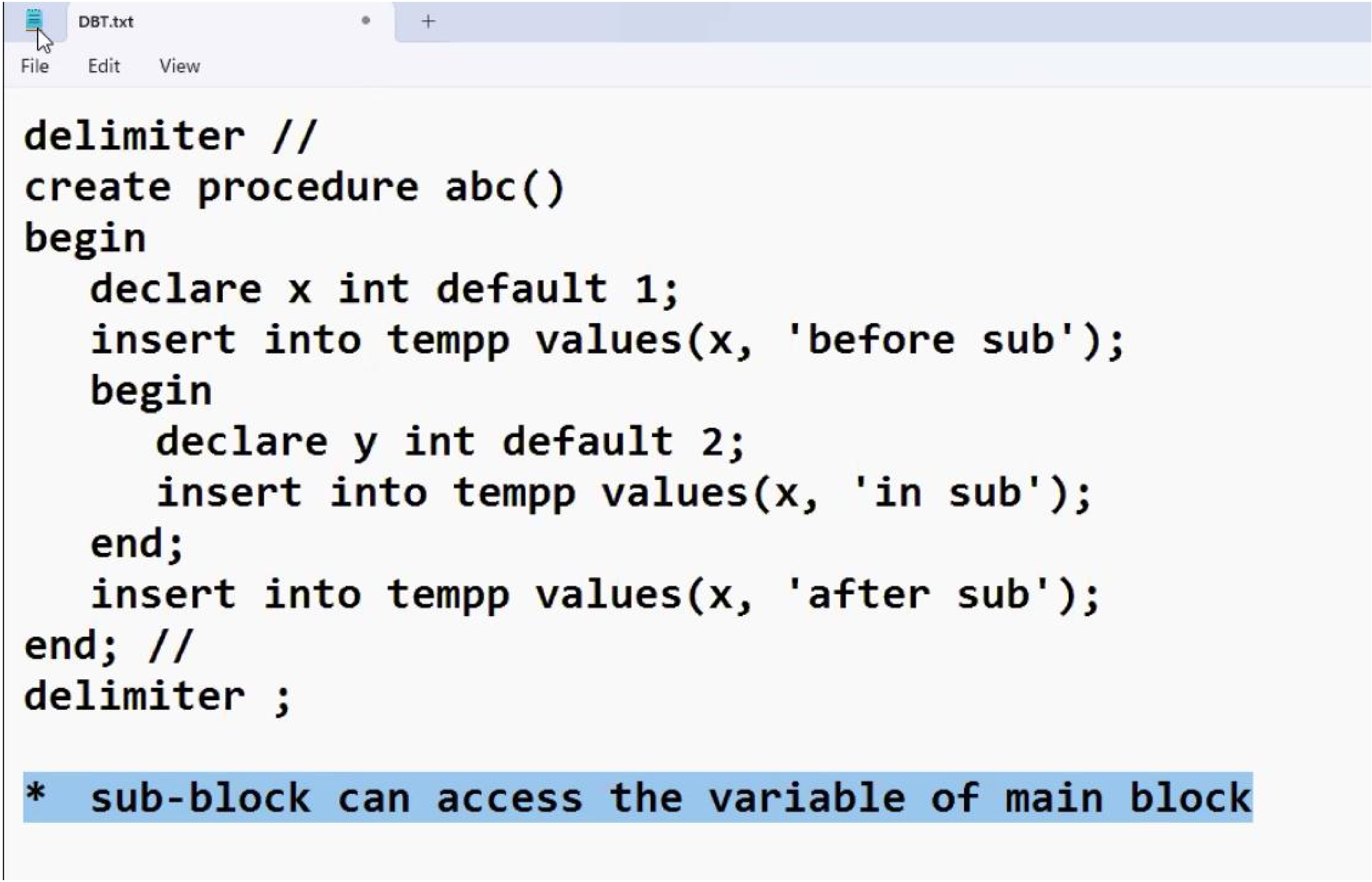


```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   repeat
      insert into tempp values(x, 'in loop');
      set x = x+1;
   until x > 5
   end repeat;
end; //
delimiter ;
```

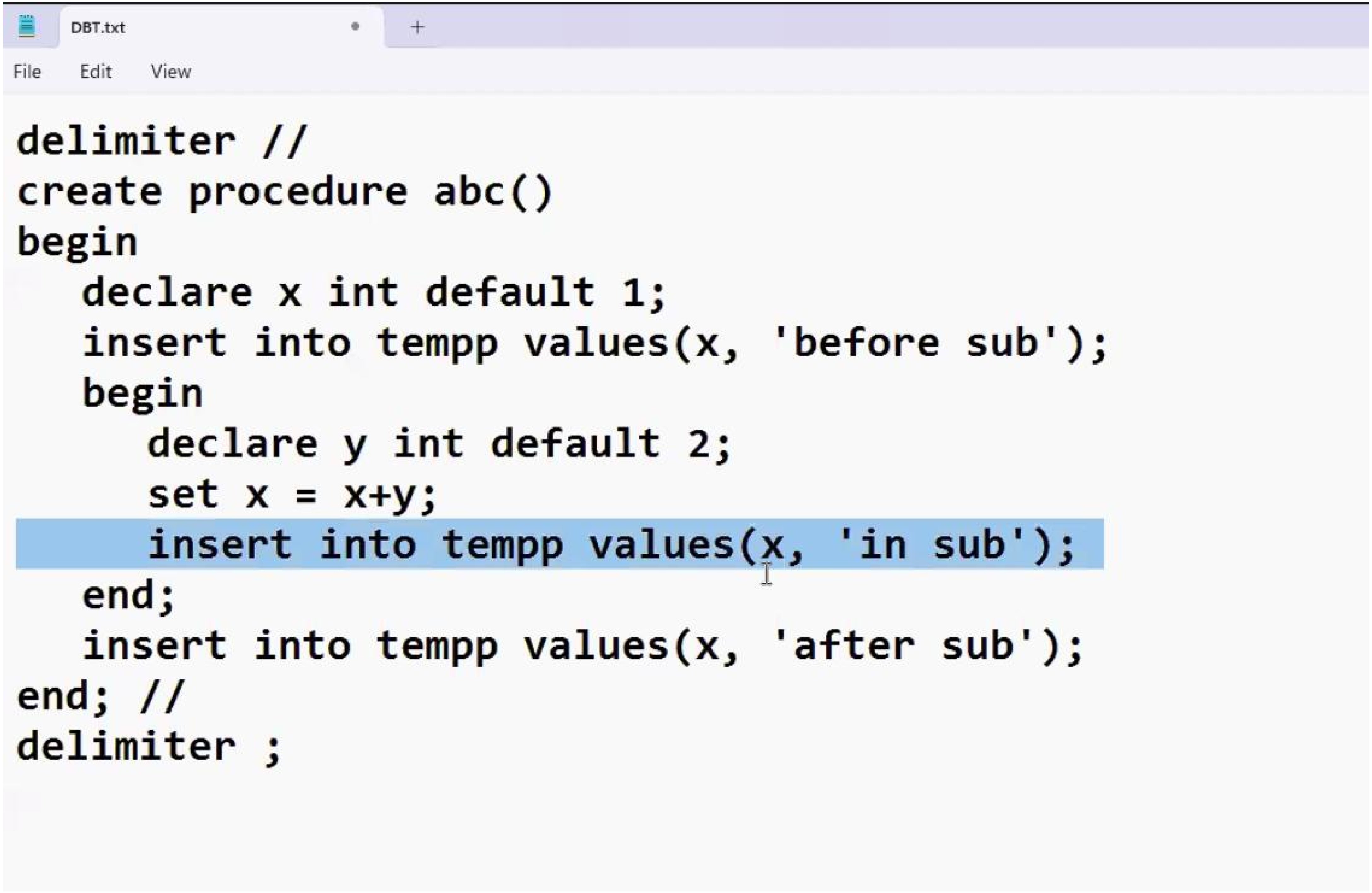


```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   insert into tempp values(x, 'before sub');
   begin
      declare y int default 2;
      insert into tempp values(y, 'in sub');
   end;
   insert into tempp values(y, 'after sub');     <- ERROR
end; //
delimiter ;
```

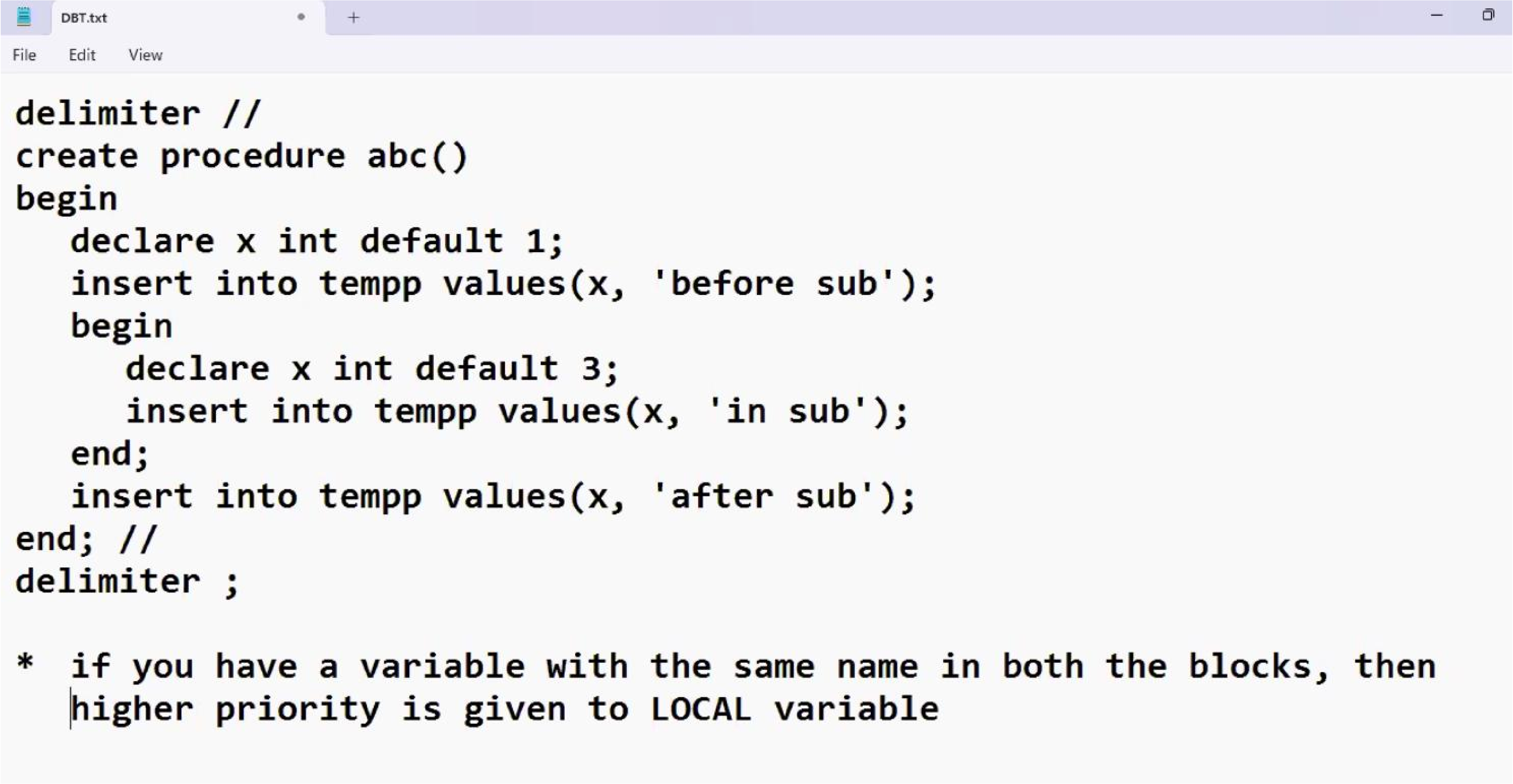
* main block cannot access the variable of sub-block



```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   insert into tempp values(x, 'before sub');
   begin
      declare y int default 2;
      insert into tempp values(x, 'in sub');
   end;
   insert into tempp values(x, 'after sub');
end; //
delimiter ;

*  sub-block can access the variable of main block
```

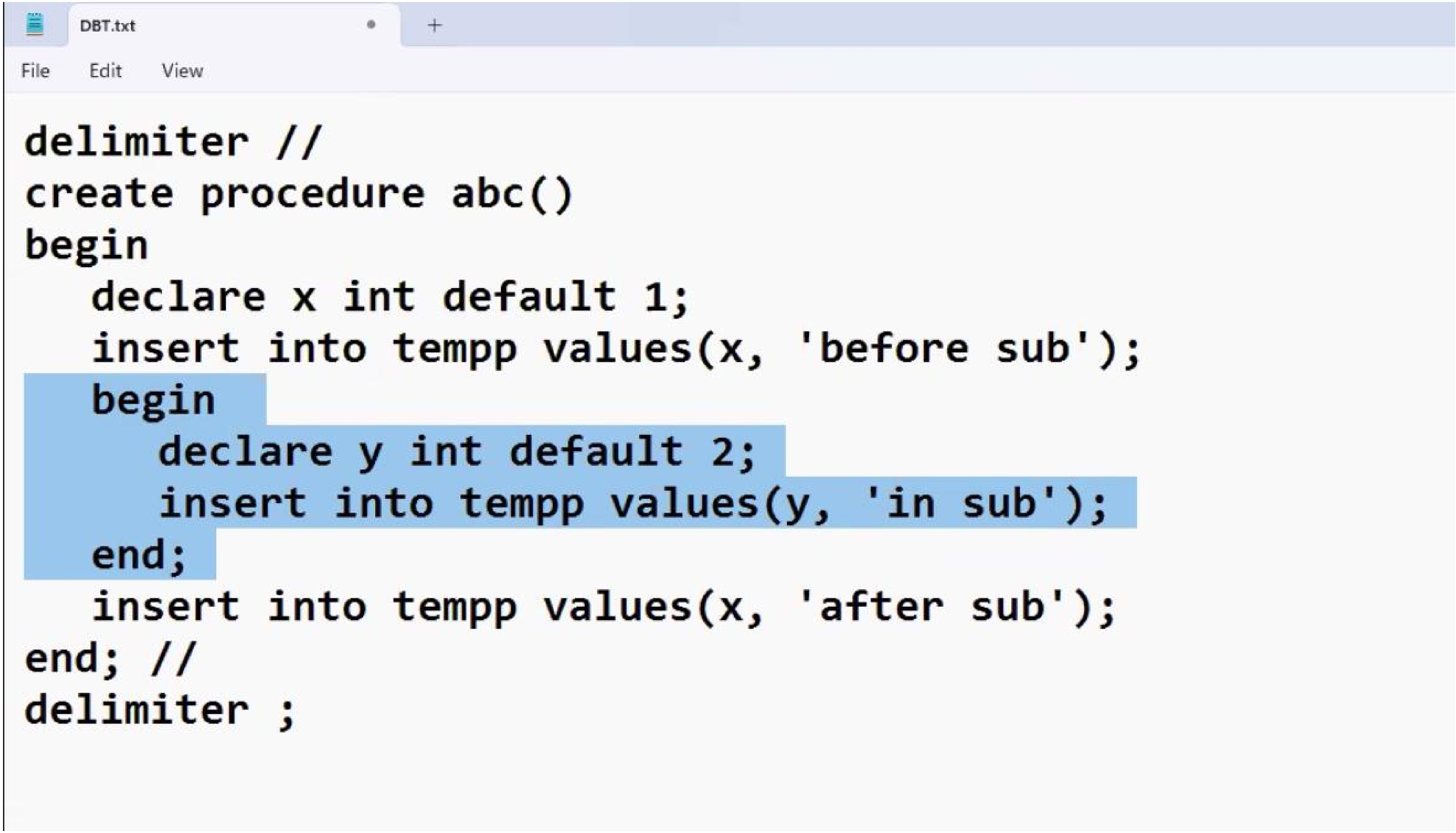


```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   insert into tempp values(x, 'before sub');
   begin
      declare y int default 2;
      set x = x+y;
      insert into tempp values(x, 'in sub');
   end;
   insert into tempp values(x, 'after sub');
end; //
delimiter ;
```



```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   insert into tempp values(x, 'before sub');
   begin
      declare x int default 3;
      insert into tempp values(x, 'in sub');
   end;
   insert into tempp values(x, 'after sub');
end; //
delimiter ;
```

* if you have a variable with the same name in both the blocks, then
  higher priority is given to LOCAL variable



```
delimiter //
create procedure abc()
begin
   declare x int default 1;
   insert into tempp values(x, 'before sub');
   begin
      declare y int default 2;
      insert into tempp values(y, 'in sub');
   end;
   insert into tempp values(x, 'after sub');
end; //
delimiter ;
```