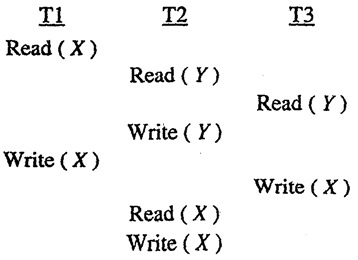
**EXPERIMENT 12**

**Implement PL/SQL Programs on Case Study 3& 6**

**(WAREHOUSE SYSTEM) & (PAINITING HIRE BUSINESS)**

**PRE-LAB**

1. Consider the following schedule for transactions T1, T2 and T3 .



State the appropriate schedules which matches the correct serialization of the above?

**Ans)**

Explanation: T1 can complete before T2 and T3 as there is no conflict between write(x) of T1 and the operations in T2 and T3 which occurs before write(x) of T1 in the above diagram   
2**.** What is displayed on the screen after I execute the following statements?

**CREATE TABLE plch\_stuff  
(  
 id INTEGER PRIMARY KEY,  
 nm VARCHAR (5) UNIQUE  
)  
/  
DECLARE l\_count PLS\_INTEGER;  
BEGIN  
INSERT INTO plch\_stuff  
VALUES (1, 'Hat');  
INSERT INTO plch\_stuff  
 VALUES (1, 'Jacket');**

**DECLARE EXIT HANDLER FOR SQLEXCEPTION  
 BEGIN  
 SELECT COUNT (\*) INTO l\_count FROM plch\_stuff;  
 PUT\_LINE (CONCAT('Rows = ' , ifnull(l\_count,'')));  
 END;  
END;  
/**

**Ans)**

An error is displayed during execution of following statements because the id is declared as primary key (Unique) but insertion of id is done with same id as ‘1’

3. Consider the transactions T1, T2, and T3 and the schedules S1 and S2 given below.

T1:r1(X);r1(Z);w1(X);w1(Z)

T2:r2(Y);r2(Z);w2(Z)

T3:r3(Y);r3(X);w3(Y)

S1: r1(X); r3(Y); r3(X); r2(Y); r2(Z); w3(Y); w2(Z); r1(Z); w1(X); w1(Z)

S2: r1(X); r3(Y); r2(Y); r3(X); r1(Z); r2(Z); w3(Y); w1(X); w2(Z); w1(Z)

Which one of the following statements about the schedules is TRUE?

(A) Only S1 is conflict-serializable (B) Only S2 is conflict-serializable

(C) Both S1 and S2 are conflict-serializable (D) Neither S1 nor S2 is conflict-serializable

**Ans)**

T1: r1(x) ; r1(z) : w1(x) ; w1(z)

T2: r2(y) ; r2(z) ; w2(z)

T3: r3(y) ; r3(x) ; w3(y)

S1: r1(x) : r3(y) : r3(x) ; r2(y) ; r2(z) ; W3(y) ; w2(z) ; r1(z) ; w1(x) ; w1(z)

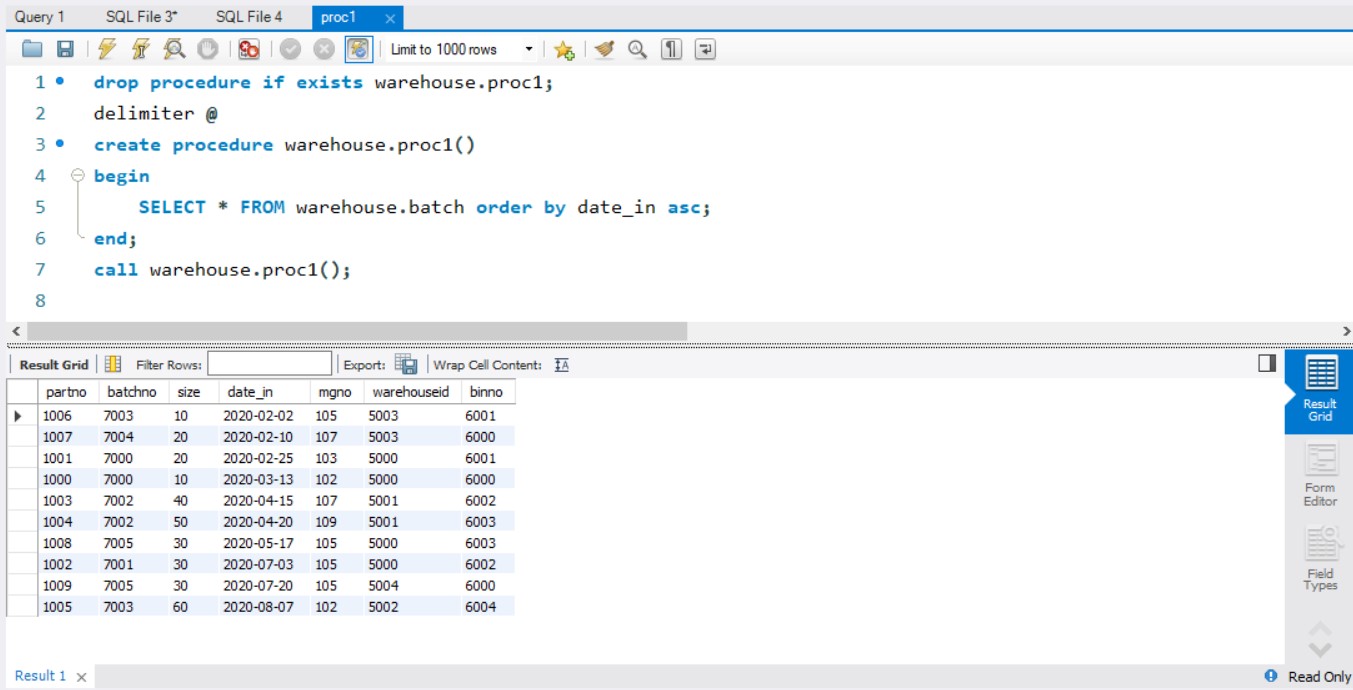
S2: r1(x) ; r3(y) ; r2(y) ; r3(x) ; r1(z); R2(z); w3(y); w1(x); w2(z) ; w1(z) ;

**Solution:** Only S1 is conflict -serializable.

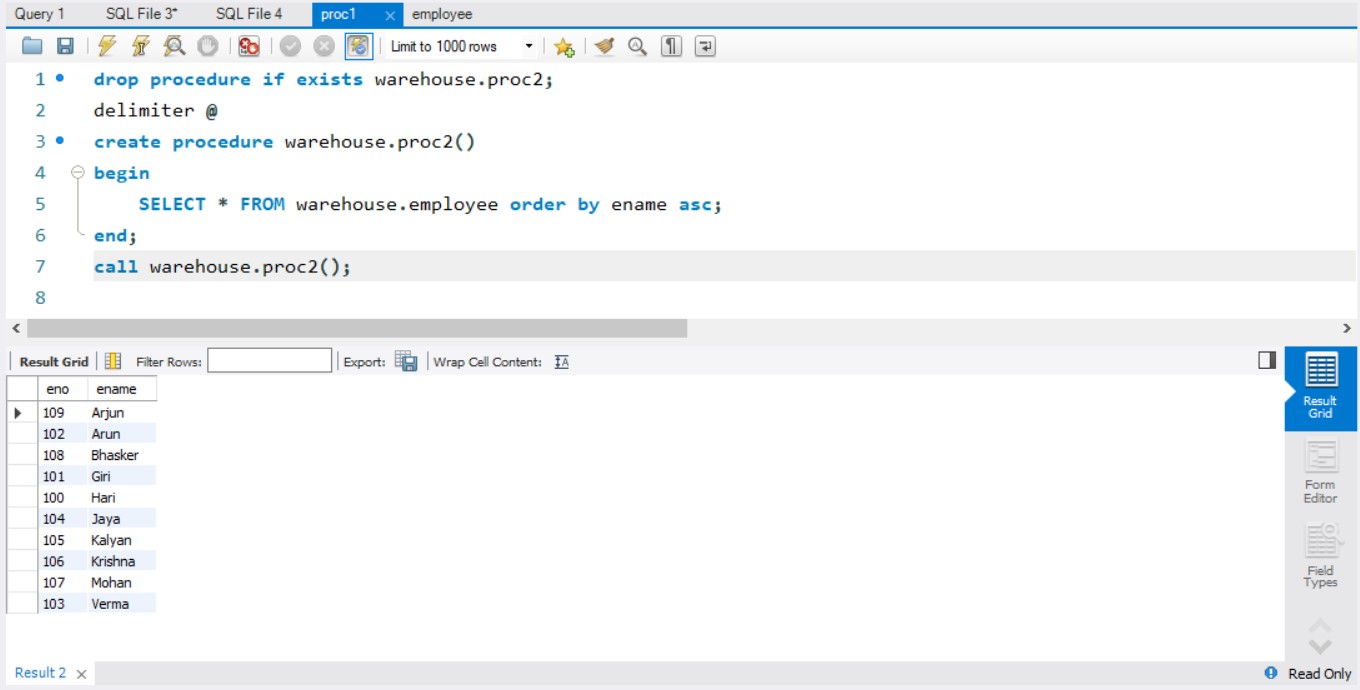
#### **IN-LAB**

**PL/SQL PROGRAMS ON WAREHOUSE SYTEM**

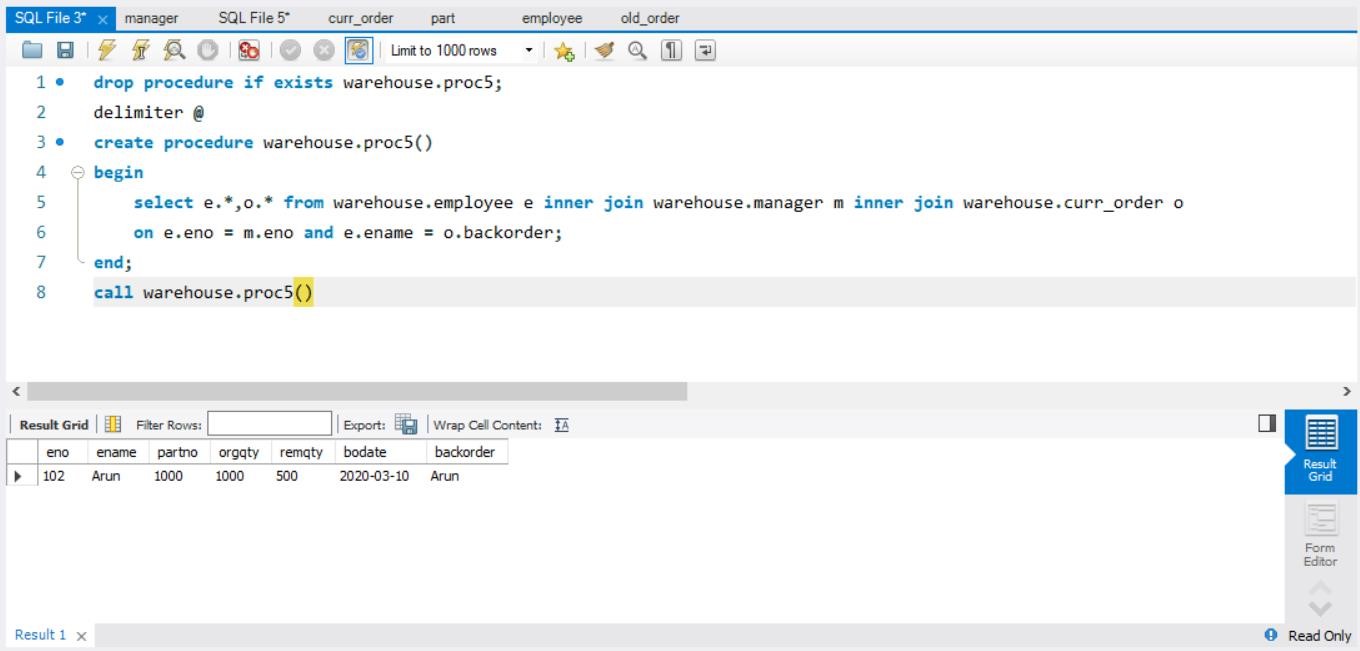
#### Write PL/SQL Program to display Batch details in ascending order of Date



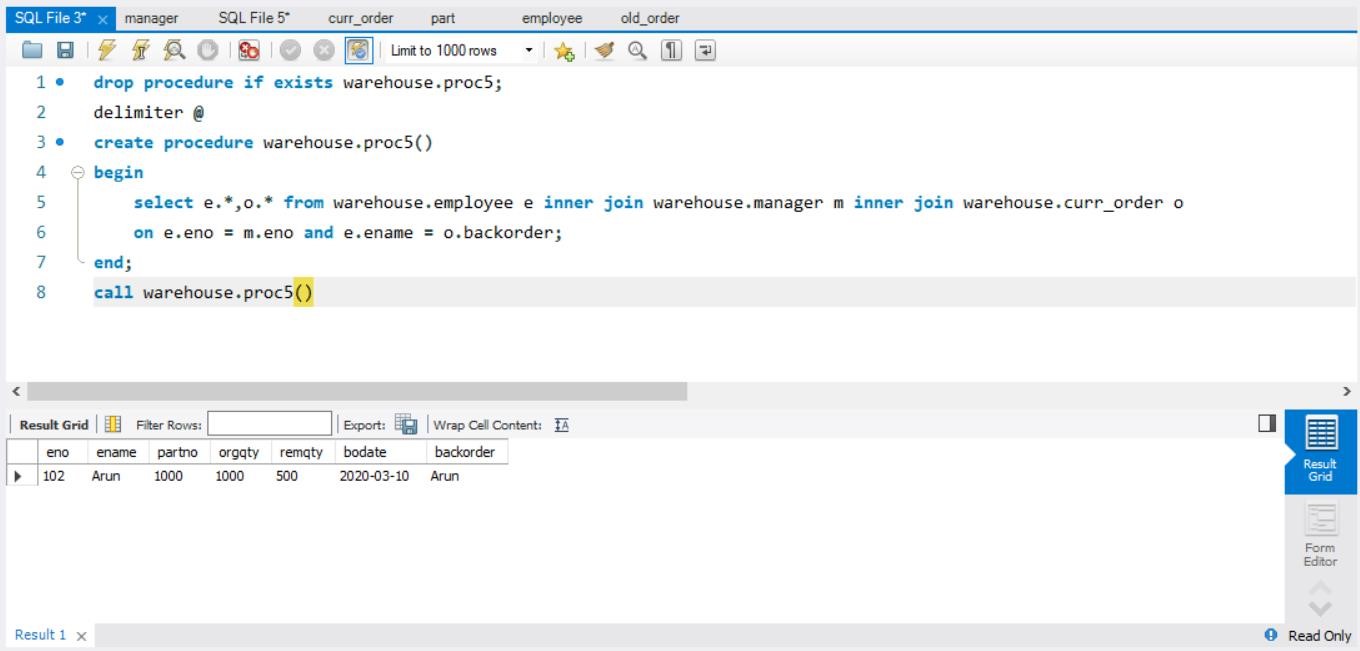
#### Write PL/SQL Program to display the names and employee\_id for all the managers. Names should be listed in alphabetic order.



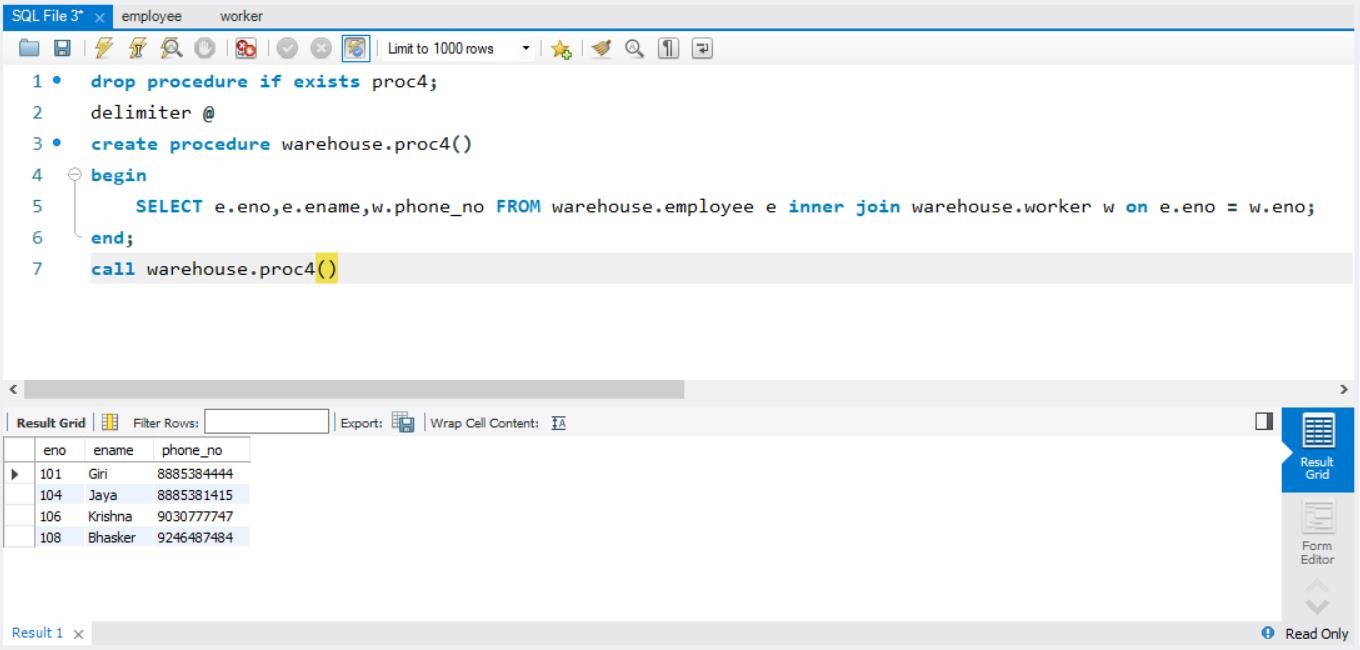
#### Write PL/SQL Program to list all current and old backorders done by the each manager. For each backorder you have to list the part no, backorder date, and fulfilled date. For current backorders, list a fulfilled date '2020-01-01'.



#### Write PL/SQL Program to list out the remaining capacity of the bin for each warehouse.

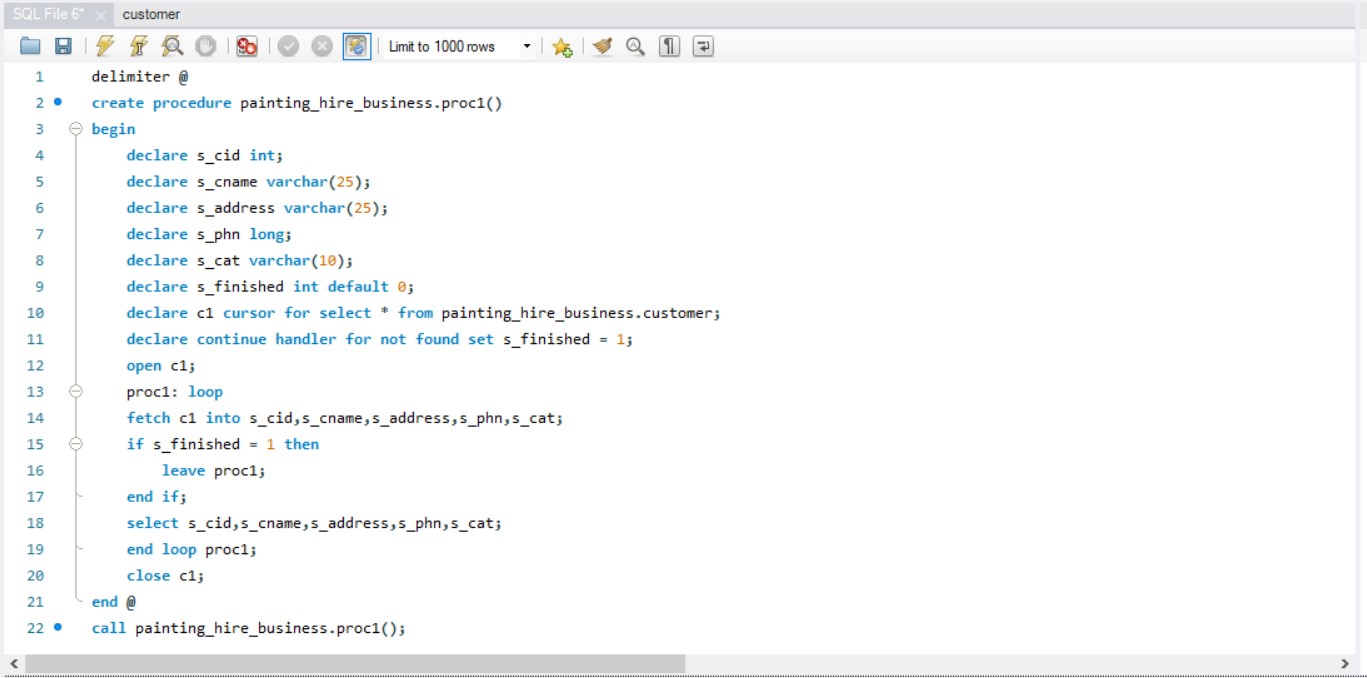


#### Write a PL/SQL Program to display all the phones and employee no for all the workers.

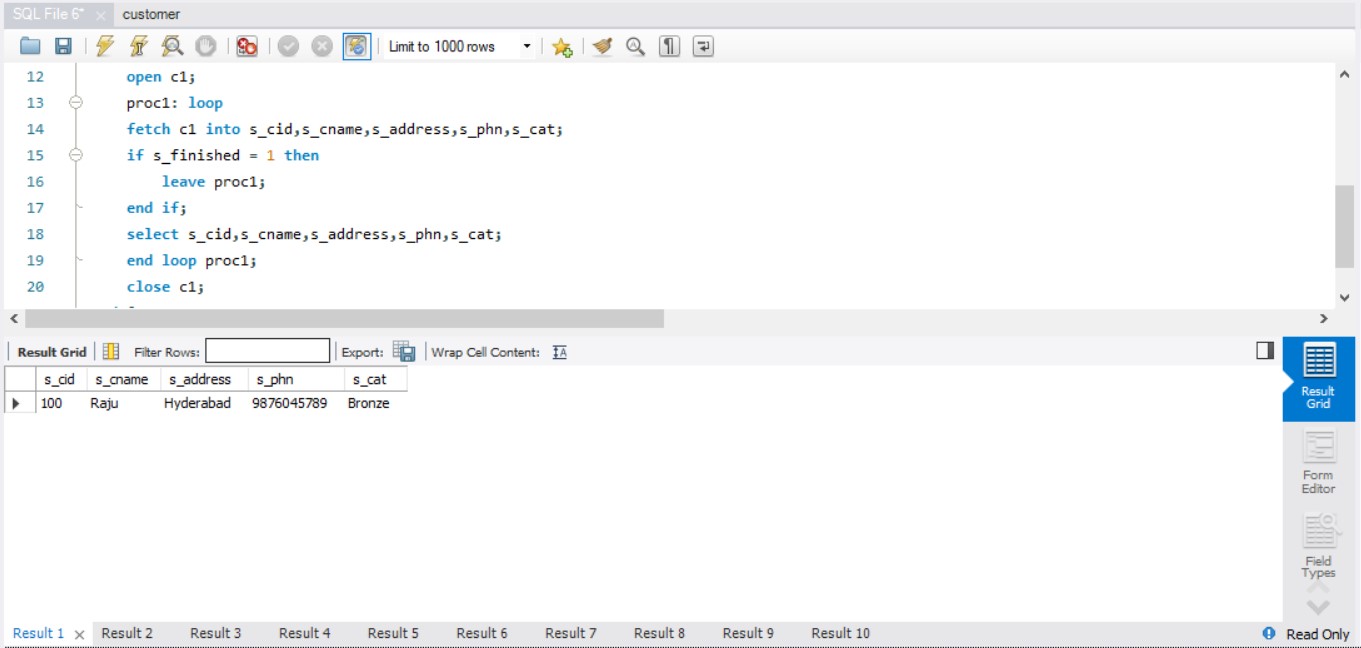


**PL/SQL PROGRAMS ON PAINTING HIRE BUSINESS**

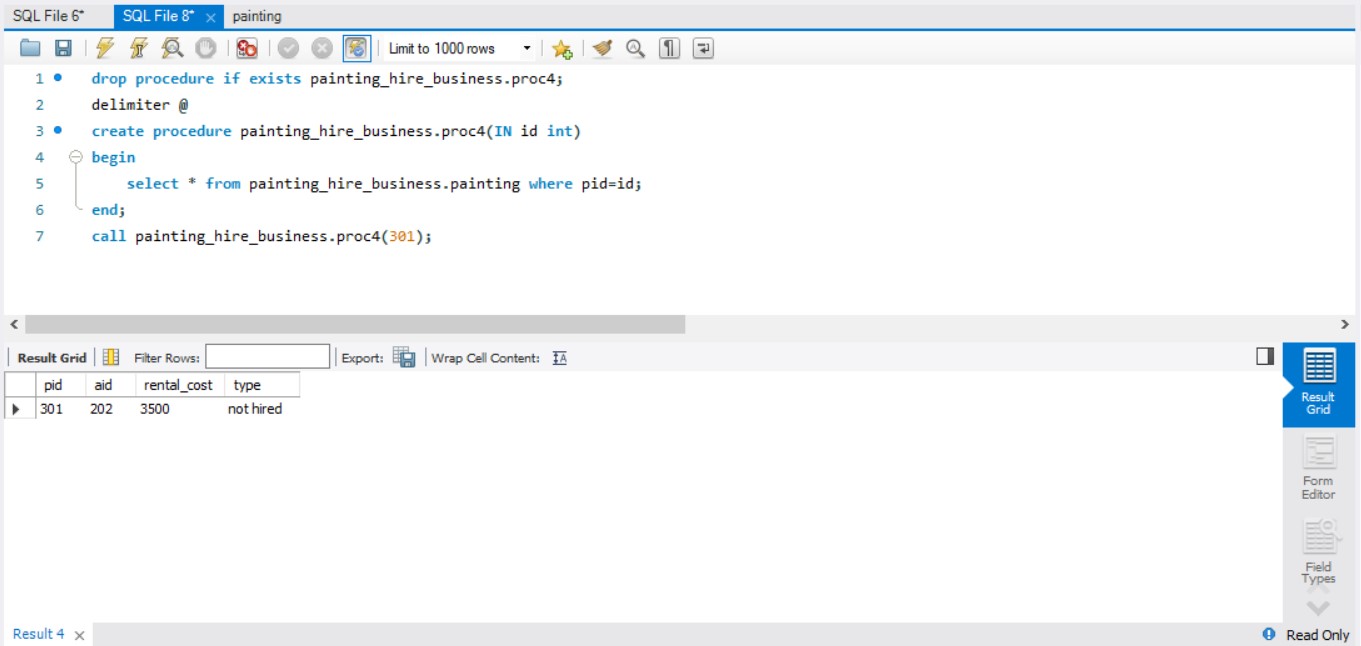
1. Create a cursor to display all the customer details



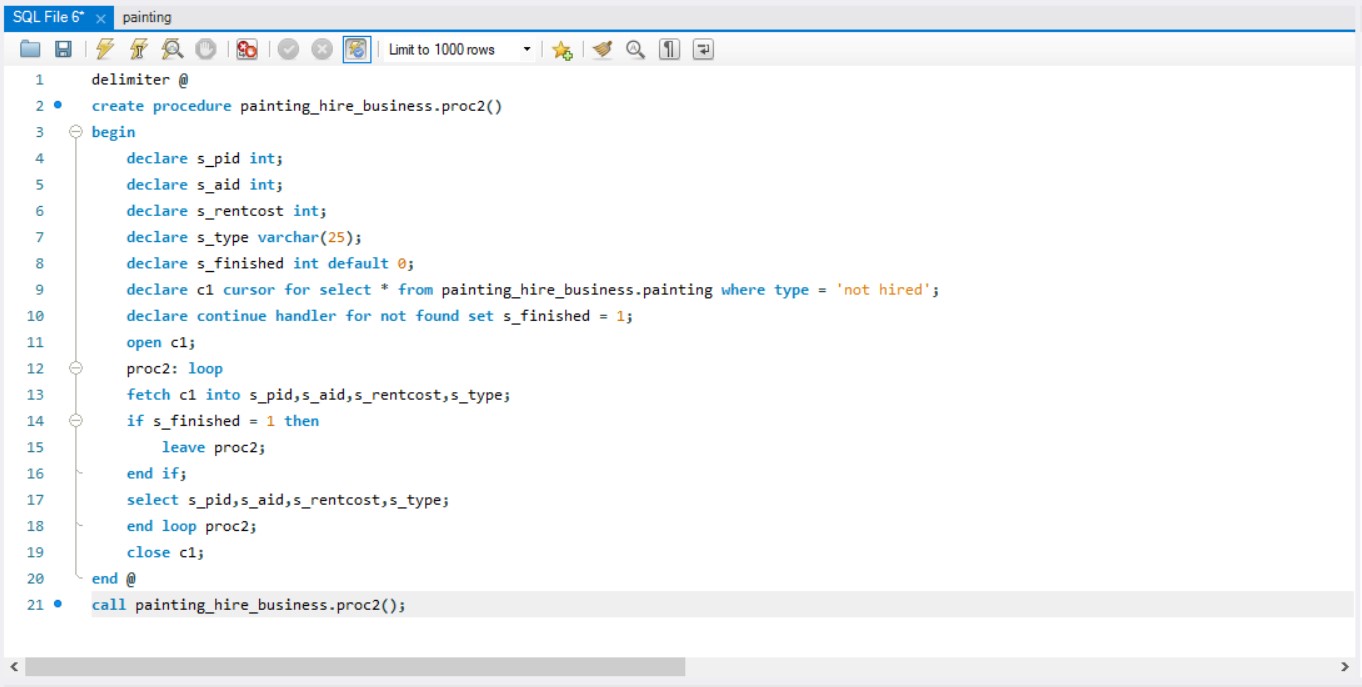
Output:



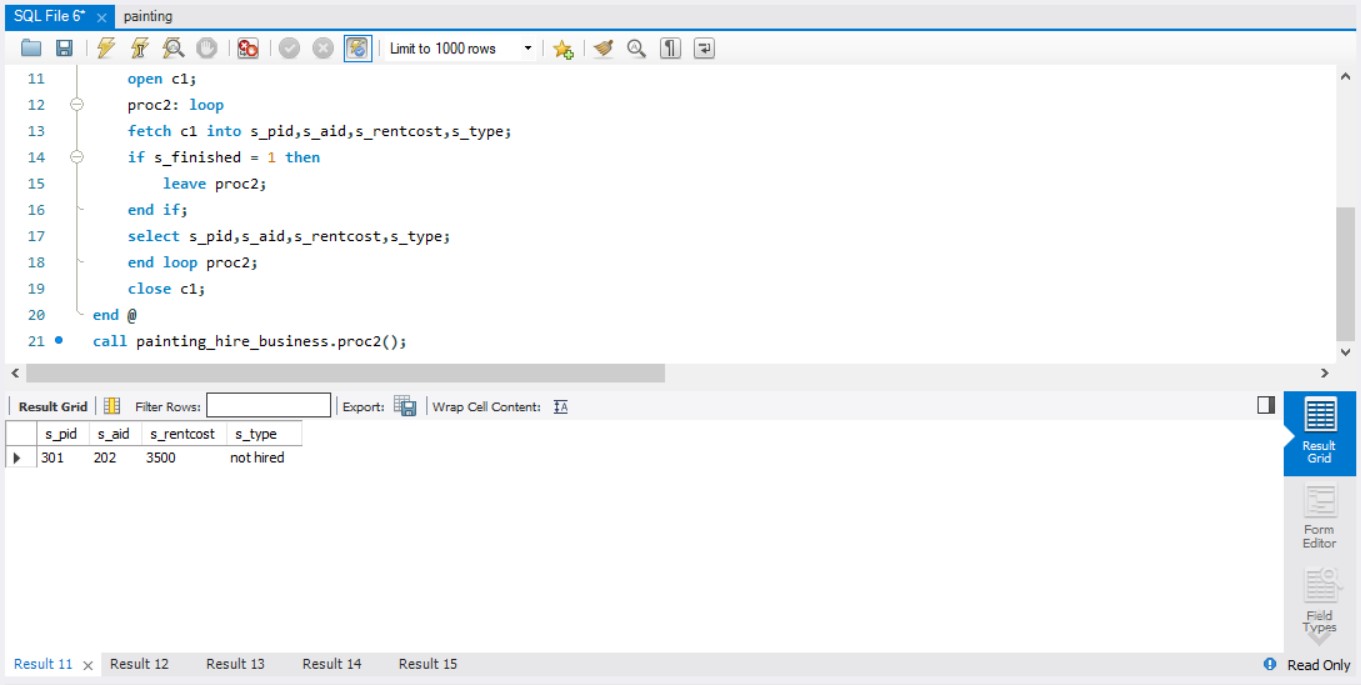
1. Create a procedure to display the required painting details



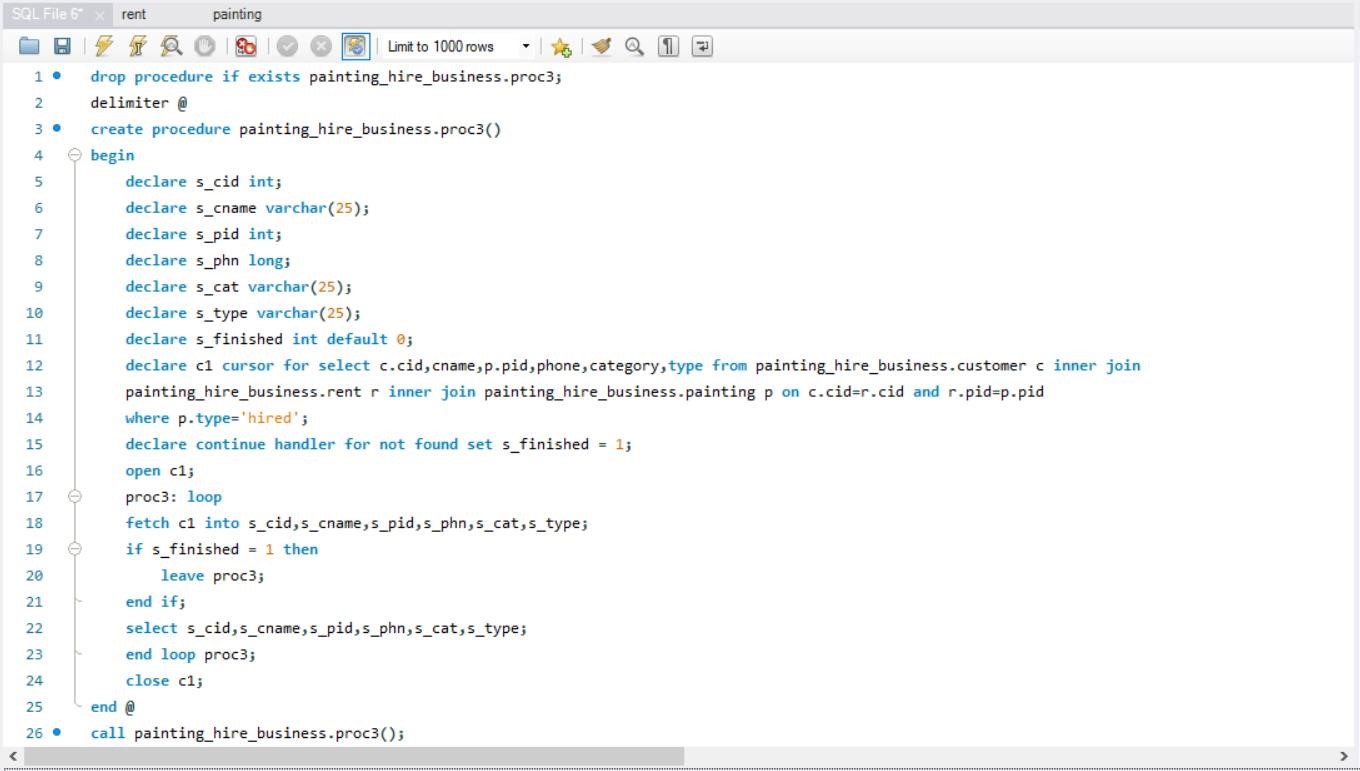
1. Create a cursor to display the painting details which are not hired



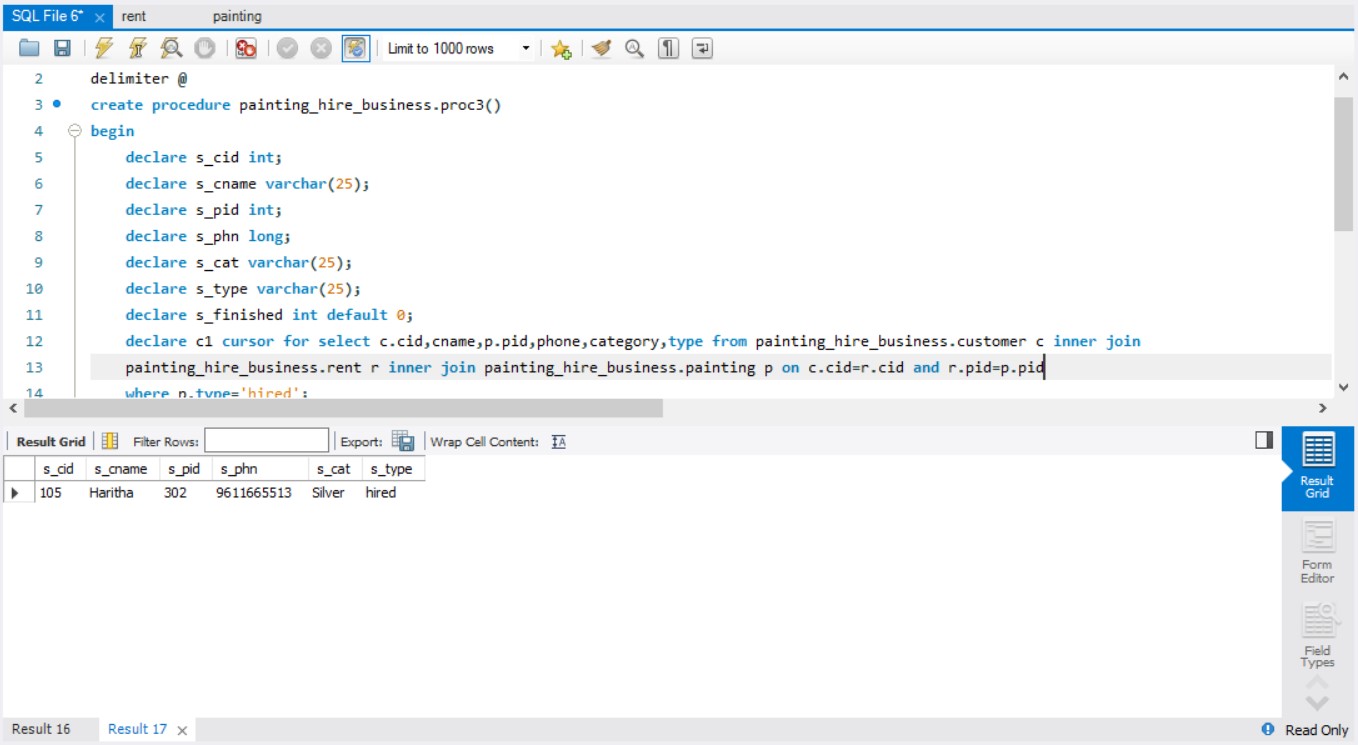
**Output:**



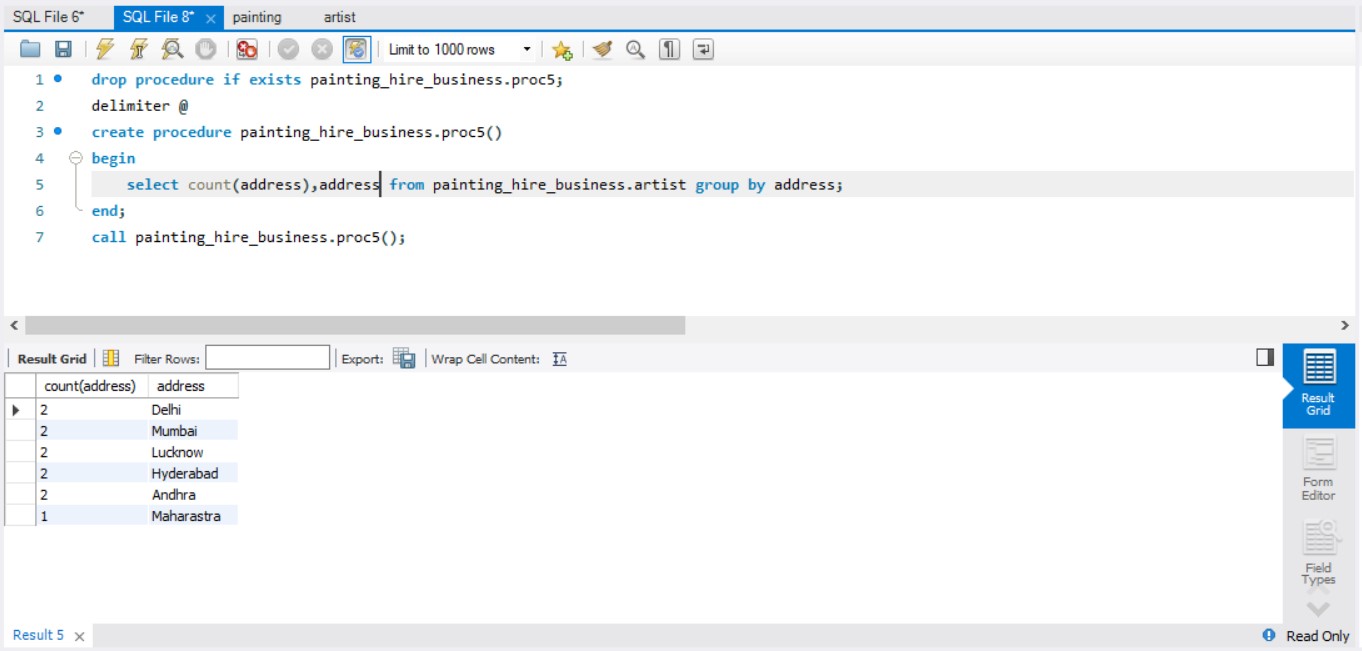
1. Create a cursor to display customer details which have been hired



**Output:**



1. Create a procedure to display the count of artists in each city



**POST-LAB**

#### Write a PL/SQL block to describe the usage of LIKE operator including wildcard characters and escape character.

**Ans)**

**Declare**

**Procedure pat\_match( test\_string varchar(10), pattern varchar(10)) is**

**Begin**

**If test\_string like pattern then.**

**Dbms\_output.put\_line(‘true’);**

**Else**

**Dbms\_output.put\_line(‘False’);**

**End If;**

**End;**

**Begin**

**Path\_match(‘blweate’ , ‘B/.a\_e’);**

**Path\_match(‘Blweate’ , ‘B/.A\_E’);**

**End;**

#### Write a PL/SQL block to show the operator precedence and parentheses in several more complex expressions.

**Ans)**

**Declare**

**Salary number :=40000;**

**Commision number :=0.15;**

**Begin**

**Dbms\_output.put\_line(‘8+20/4=’ || (8+20/4));**

**Dbms\_output.put\_line(‘20/4+8=’ || (20/4+8));**

**Dbms\_output.put\_line(‘7+9/3=’ || (7+9/3));**

**Dbms\_output.put\_line(‘(7+9)/3=’ || ((7+9)/3));**

**Dbms\_output.put\_line(‘(salary \*0.08) + (commission \*0.12)=’ || ((salary \*0.08)+((commission \* 0.12)));**

**Dbms\_output.put\_line(‘30+(30/6+(15-8))=’ ||(30+(30/6+(15-8))));**

**Dbms\_output.put\_line(‘salary \*0.08 + commission \* 0.12 =’ || (salary \*0.08 + commission \*0.12));**

**End;**

**/**

1. Write a PL/SQL block to create a procedure using the "IS [NOT] NULL Operator" and show AND operator returns TRUE if and only if both operands are TRUE.

**Ans)**

**Begin**

**If bool\_val is null then**

**Dbms\_output.put\_line(boo\_name || ‘=Null’);**

**Else If bool\_val =True then**

**Dbms\_output.put\_line(boo\_name || ‘=True’);**

**Else**

**Dbms\_output.put\_line(boo\_name || ‘=False’);**

**End if;**

**End;**

**/**

1. Write a PL/SQL block to show an invalid case-insensitive reference to a quoted and without quoted user-defined identifier.

**Ans)**

**Declare**

**“Welcome varchar(10):= ‘welcome’;**

**Begin**

**Dbms\_output.put\_line(“welcome”);**

**End;**

**/**

1. Write a PL/SQL block to explain single and multiline comments

**Ans)**

**Declare**

**Condition Boolean;**

**Pi number :=3.145;**

**Radius number :=10;**

**Area number ;**

**Begin**

**If 2+2 =4 then**

**Condition = true;**

**End if;**

**Area = pi \*radius\*2;**

**Dbms\_output.put\_line(‘The area of circle is :’ || area);**

**End;**

**/**

1. Consider the following transaction involving two bank accounts x and y. read(x); x : = x–50; write(x); read(y); y:=y+50; write(y) The constraint that the sum of the accounts x and y should remain constant is that of

(A) Atomicity

(B) Consistency

(C) Isolation

(D) Durability

**Ans)**

**Option (B) Consistency**

**Explanation : Consistency in database system refer to the requirement that any given database transaction must only change affected data in allowed ways , that is sum of x and y must not change**

1. Consider a simple checkpointing protocol and the following set of operations in the log. (start, T4); (write, T4, y, 2, 3); (start, T1); (commit, T4); (write, T1, z, 5, 7);(checkpoint); (start, T2); (write, T2, x, 1, 9); (commit, T2); (start, T3), (write, T3, z, 7, 2);If a crash happens now the system tries to recover using both undo and redo operations, what are the contents of the undo list and the redo list.

(A) Undo: T3, T1; Redo: T2

(B) Undo: T3, T1; Redo: T2, T4

(C) Undo: none; Redo; T2, T4, T3, T1

(D) Undo: T3, T1, T4; Redo: T2

**Ans)**

**Option (A) Undo :T3, T1 ; Redo:T2.**

8) Amongst the ACID properties of a transaction, the 'Durability' property requires that the changes made to the database by a successful transaction persist

A. Except in case of an Operating system crash

B. Expect in case of Disk Crash

C. Expect in case of a power failure

D. Always, even if there is a failure of any kind

**Ans)**

**Option (D) Always ,even if there is a failure of any kind**