CSE2004 – Database Management Systems Lab
Cycle Sheet Submission

Register Number : 19BCE2074

Name : KULVIR SINGH

Slot : L45 + L46

Mobile Number : 7541097627

RDBMS used : MYSQL/MYSQL WORKBENCH

Cycle Sheet No. : 03

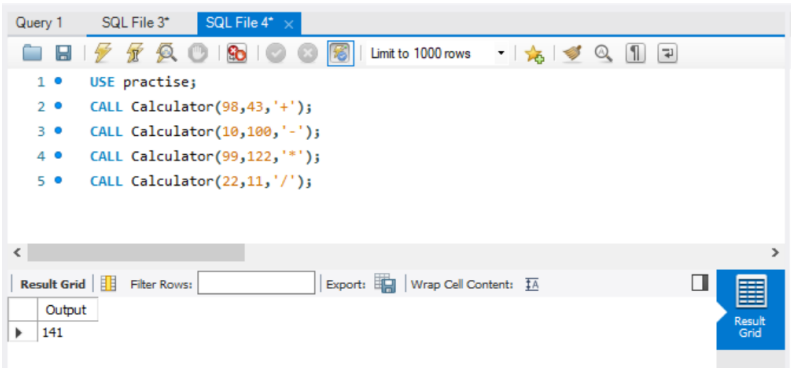
1. Write a PL/SQL program to implement a simple calculator.

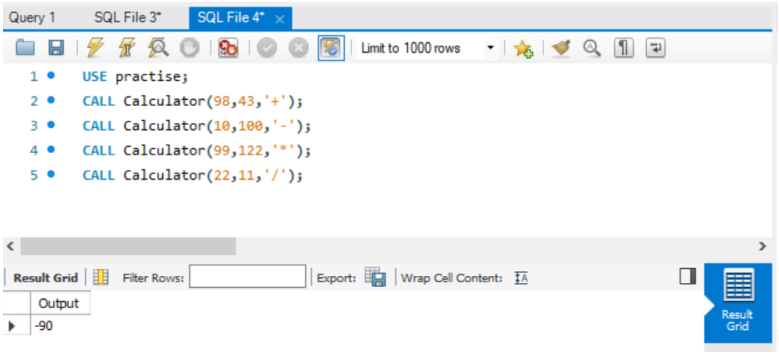
CODE:

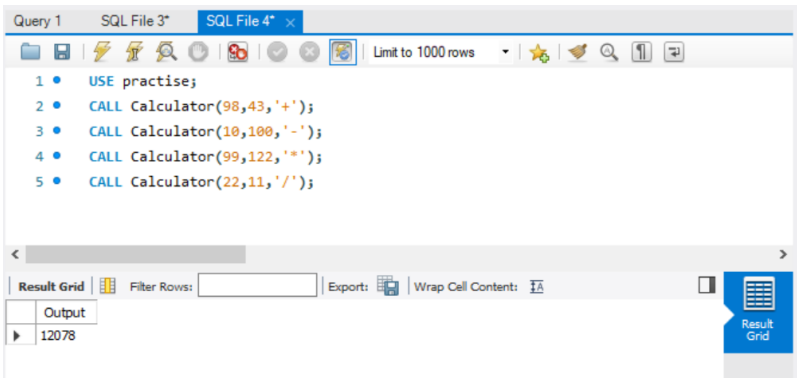
```
USE practise;
DELIMITER //

CREATE PROCEDURE Calculator(
   IN a INT(3),
   IN b INT(3),
   IN op CHAR(1)
)
BEGIN

DECLARE Output VARCHAR(255);

        IF op = '+'
                THEN
                        SET Output = a + b;
        ELSEIF op = '-'
                THEN
                        SET Output = a-b;
        ELSEIF op = '*'
                THEN
                        SET Output = a*b;
        ELSEIF op = '/'
                THEN
                        IF b = 0
                                THEN
```

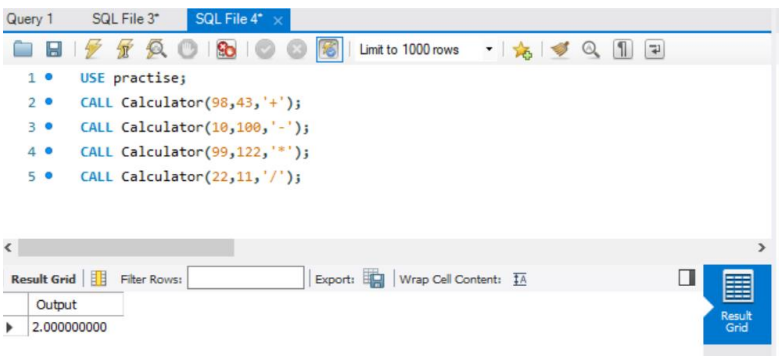
```
                                        SET Output = 'INFINITE';
                        ELSE
                                SET Output = a/b;
                        END IF;
        ELSE
                SET Output = 'INVALID';
END IF;

SELECT Output;

END //

DELIMITER ;
```

OUTPUT SCREENSHOT:







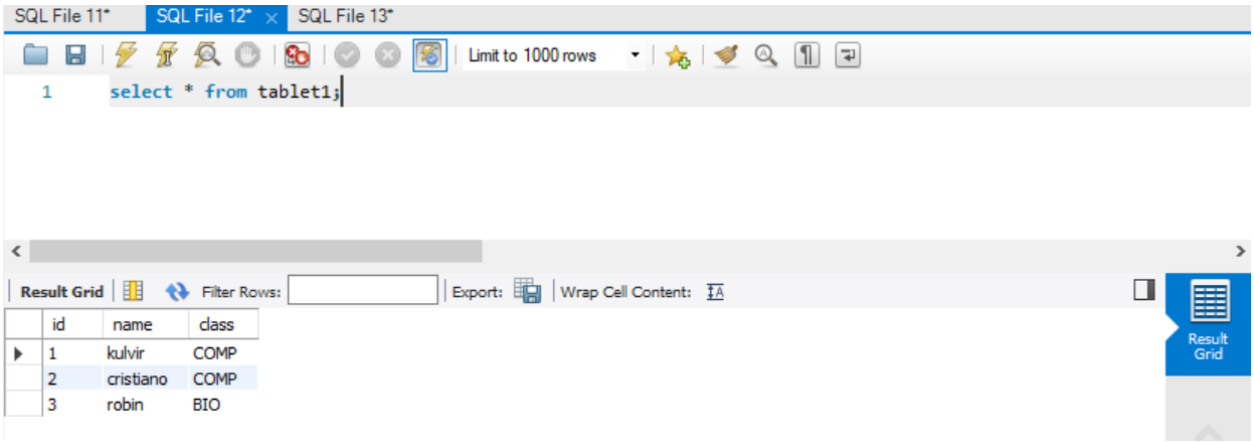


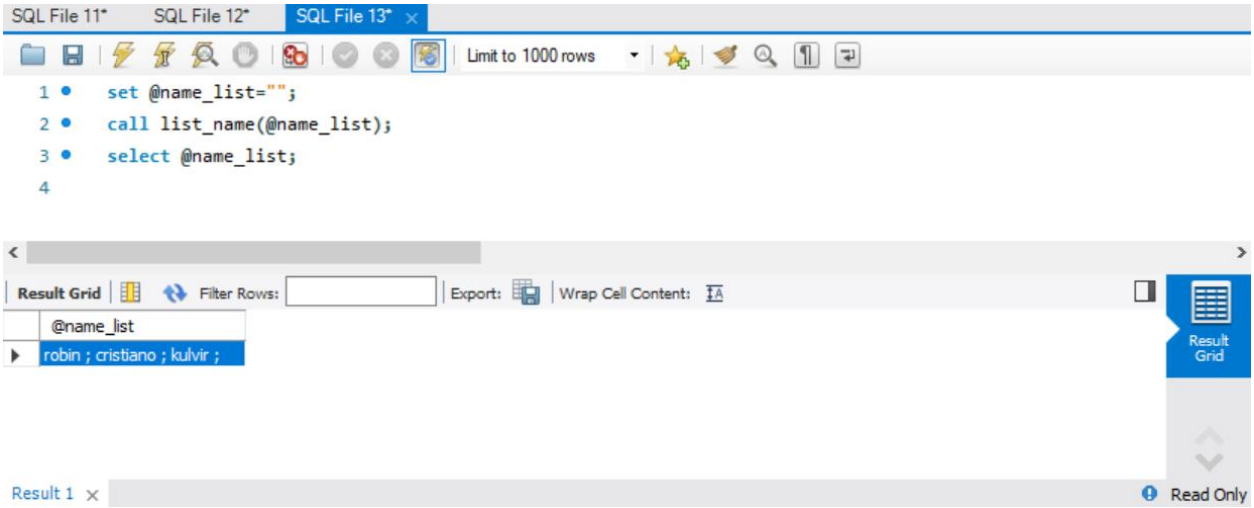
2. Write a PL/SQL program to practice reading the record from a table into local variables using different data types and %TYPE and display the same using locally declared variables.

CODE:

```
DELIMITER //
create procedure list_name(inout name_list varchar(255))
begin
declare is_done int default 0;
declare s_name varchar(255) default "";
declare stud_cursor CURSOR for
select name from tablet1;
declare continue handler for not found set is_done =1;
open stud_cursor;
get_list: LOOP
fetch stud_cursor into s_name;
if is_done=1 then leave get_list;
end if;
set name_list = concat(s_name, " ; ", name_list );
end loop get_list;
close stud_cursor;
end;//
```

OUTPUT SCREENSHOT :





3. Write a PL/SQL program to find the number of doctors in a given department with a given qualification (read values for department and qualification from user during runtime). If number is more than the number of doctors in that department with other qualifications then display 'Well qualified' else 'Qualified'.

CODE:

```
use hospital;

declare

dt_name varchar(255);

qual varchar(255);

qual_doct_cnt int;

non_qual_doct_cnt int;

begin

select count(1) into qual_doct_cnt from doctor inner join department on doctor.dept_no =
department.dept_no where department.dept_name = dt_name and qualification = qual;


select count(1) into non_qual_doct_cnt from doctor inner join department on doctor.dept_no =
department.dept_no where department.dept_name = dt_name and qualification <> qual;
```

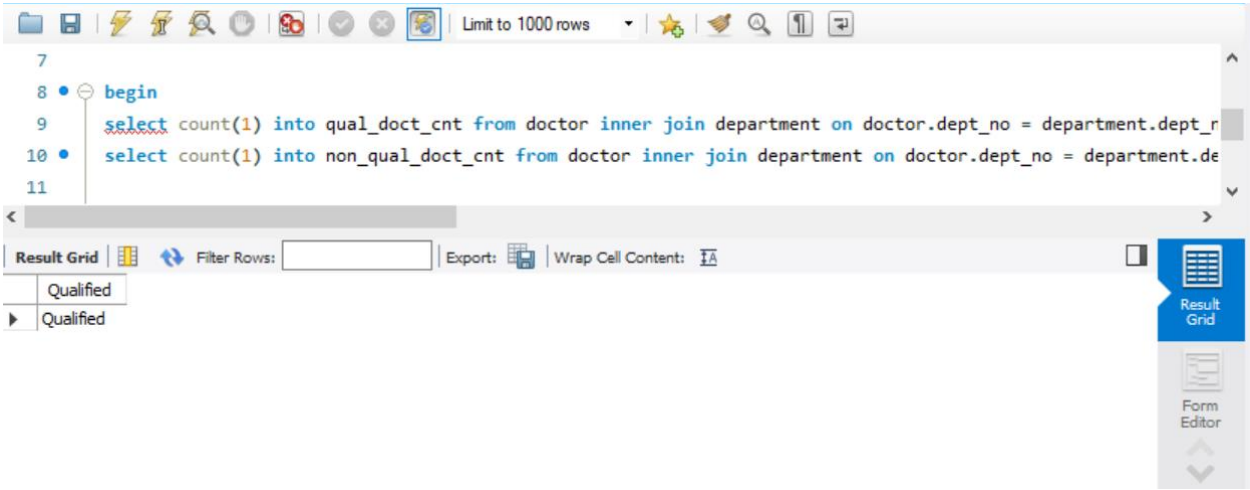
if qual_doct_cnt>non_qual_doct_cnt then select 'Well Qualified';

else select 'Qualified';

end if;

end;

OUTPUT SCREENSHOT:



4. Write a PL/SQL program to insert records into any of the tables in your database.

CODE:

DELIMITER //

CREATE PROCEDURE Insert_Records(

IN medicine_name VARCHAR(255),

IN brand_name VARCHAR(50),

IN dosage VARCHAR(40),

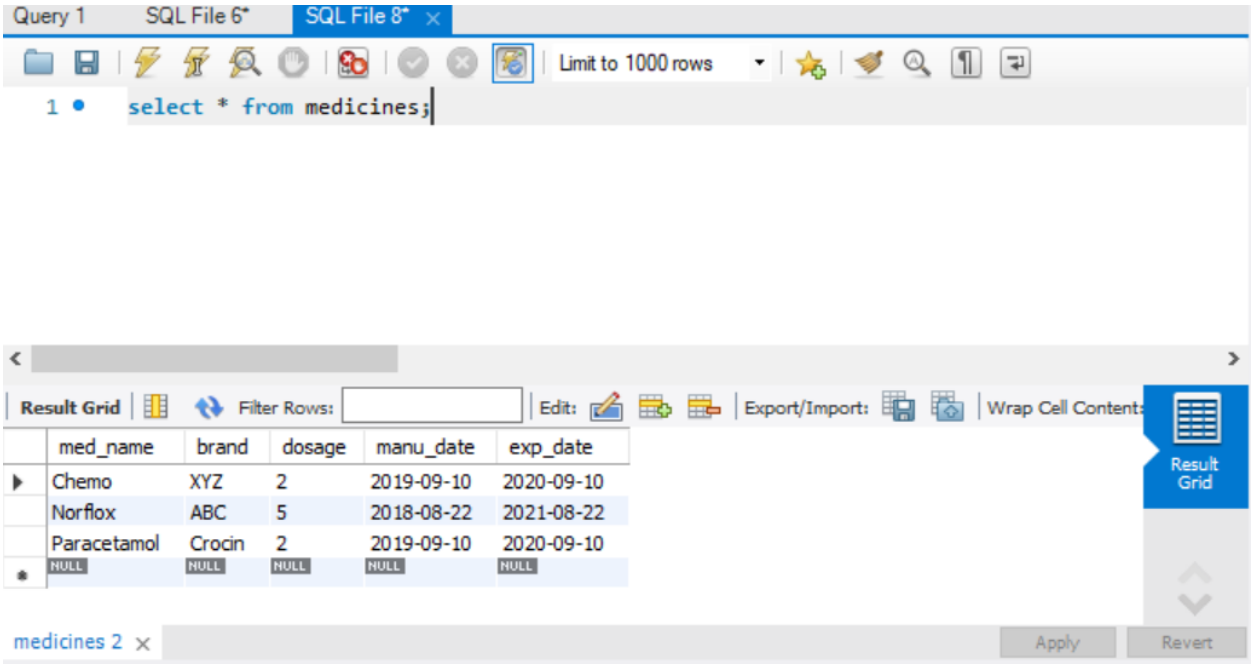
IN manu_date DATE,

IN exp_date DATE

)

```
BEGIN
INSERT INTO medicines()
VALUES(medicine_name, brand_name, dosage, manu_date, exp_date);
END //
DELIMITER ;

CALL Insert_Records('Norflox','ABC','5','2018-08-22','2021-08-22')
```

OUTPUT SCREENSHOT:



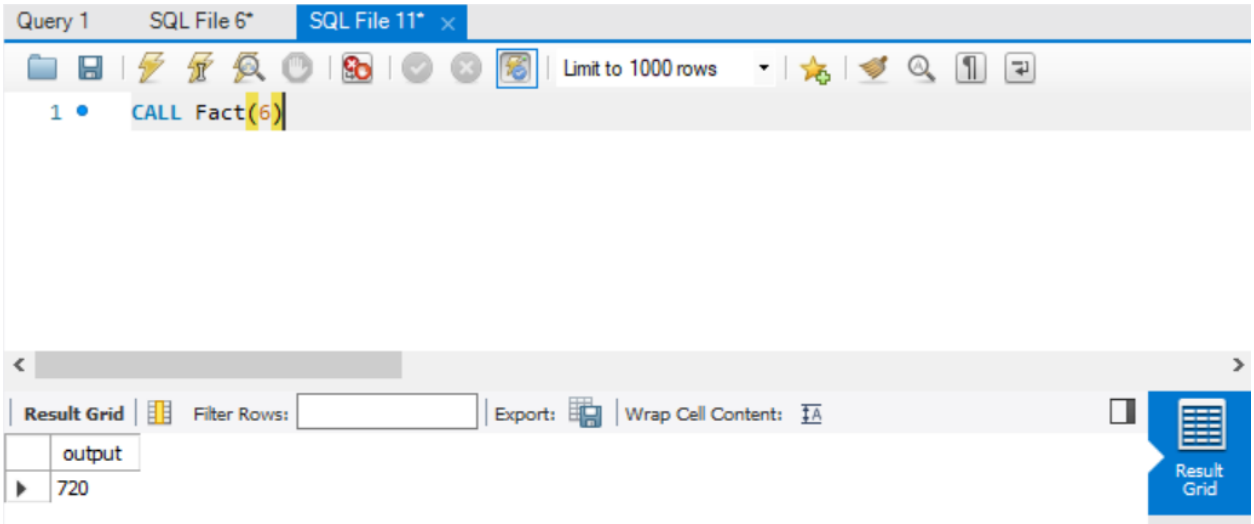
5. Create a function to find the factorial of a given number.

CODE:

```
DELIMITER //
CREATE PROCEDURE Fact(IN n INT(3))
BEGIN
        IF n < 0
```

```
            THEN
                SET @f = 'Negative factorial not defined';
        ELSEIF n = 0
            THEN
                SET @f = 1;
        ELSE
            SET @c = 1;
            SET @f = 1;

            WHILE (@c < n + 1)
                DO
                    SET @f = @f* @c;
                    SET @c = @c + 1;
            END WHILE;
        END IF;
SELECT @f AS output;
END //
DELIMITER ;
```

OUTPUT SCREENSHOT:

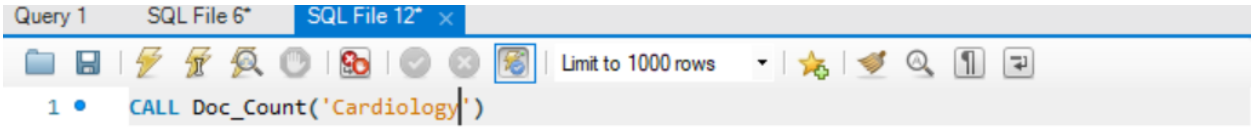


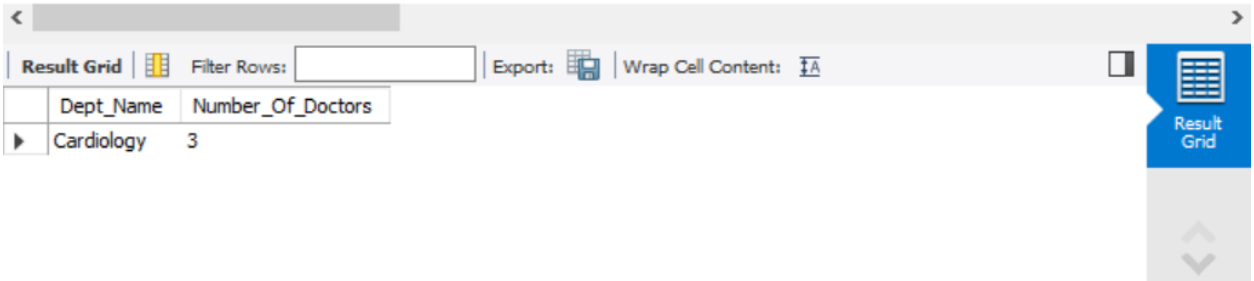
6. Create a function DOC_COUNT to find the number of doctors in the given department. Use the department name as the input parameter for the function.

CODE:

```
use hospital
DELIMITER //
CREATE PROCEDURE Doc_Count(          IN Department_Name VARCHAR(255))
BEGIN
SELECT Dept_Name, COUNT(Doc_ID) AS Number_Of_Doctors FROM department AS de
CROSS JOIN doctor AS doc
        ON de.Dept_No = doc.Dept_No
WHERE Dept_Name = Department_Name
GROUP BY Dept_Name;
END //
DELIMITER ;
```

OUTPUT SCREENSHOT:





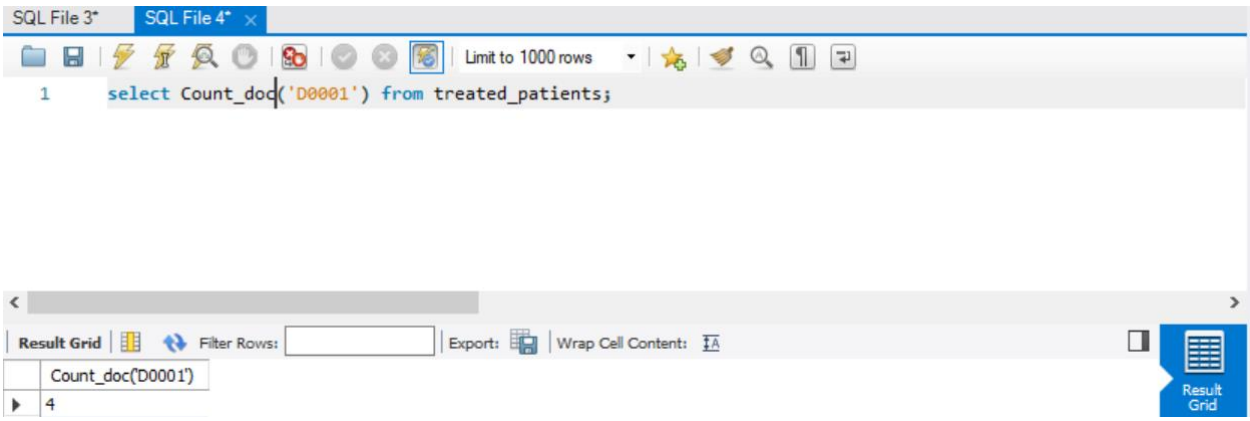
Functions and Procedures:

1. Write a PL/SQL stored function COUNT_DOC to count the number of doctors who have treated at least 100 patients if given a doctor id as input parameter.

CODE:

```
use hospital;

DELIMITER //

CREATE FUNCTION Count_Doc(doc_id varchar(5)) returns int

deterministic

begin

return(select count(doc_id) from treated_patients where pat_no>=100);

end//

DELIMITER ;
```

OUTPUT SCREENSHOT:



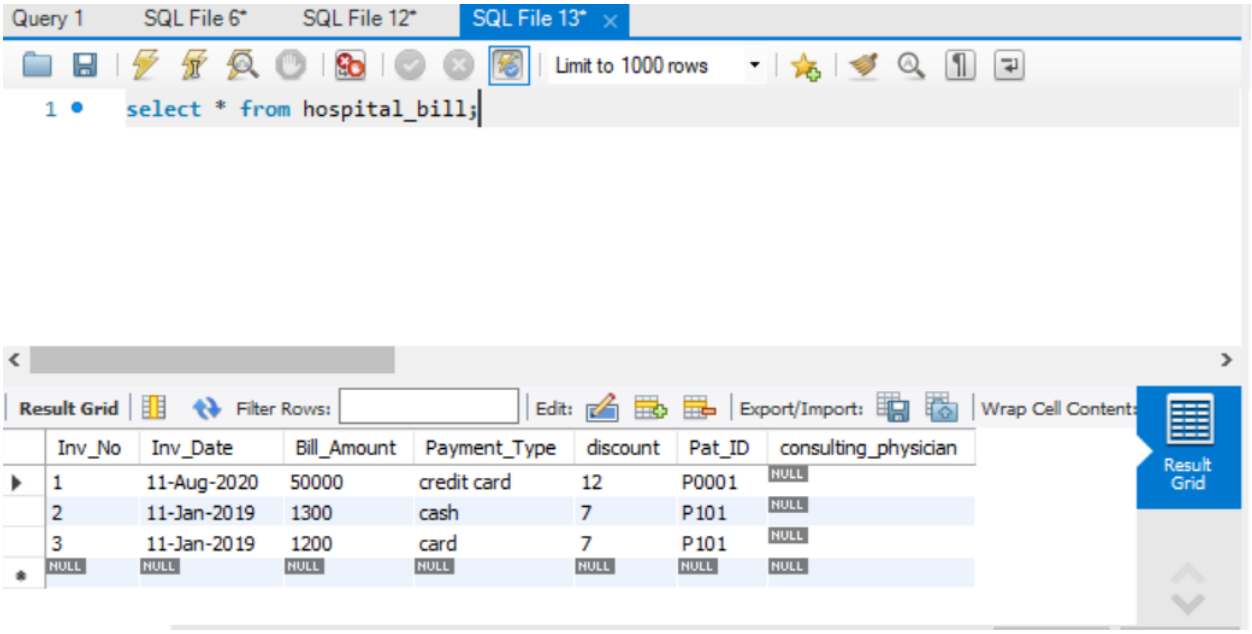
2. Write a PL/SQL stored procedure to adjust the payment type of hospital bills to CASH if the patient id and amount details given as input.

CODE:

```
use hospital;

DELIMITER //
```

```
CREATE PROCEDURE update_payment_type(IN Patient_Id VARCHAR(5), IN Invoice_No INT)
BEGIN
UPDATE hospital_bill
SET Payment_Type = 'Cash'
WHERE Inv_No = Invoice_No AND Pat_Id = Patient_Id;
END//
DELIMITER ;
```

OUTPUT SCREENSHOT:



Inv_No	Inv_Date	Bill_Amount	Payment_Type	discount	Pat_ID	consulting_physician
1	11-Aug-2020	50000	credit card	12	P0001	NULL
2	11-Jan-2019	1300	cash	7	P101	NULL
3	11-Jan-2019	1200	card	7	P101	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query 1 | SQL File 6* | SQL File 12* | SQL File 13*

Limit to 1000 rows

```
CALL update_payment_type('P0001',1);
select * from hospital_bill;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Inv_No	Inv_Date	Bill_Amount	Payment_Type	discount	Pat_ID	consulting_physician
1	11-Aug-2020	50000	Cash	12	P0001	NULL
2	11-Jan-2019	1300	cash	7	P101	NULL
3	11-Jan-2019	1200	card	7	P101	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Result Grid

Triggers:

Create a table EMP_SALARY with attributes ID, Basic, DA, HRA, Deduction, Net_Salary. Here, ID refers the Staff_ID of staff table. Treat 'Net_Salary' as a derived attribute and don't insert a value through insert operation. The value for Net Salary can be calculated as follows;

Net_Salary = Basic + DA + HRA – Deduction

1. Write a Trigger to perform the following; whenever new staff is recruited and a designation is assigned, insert an appropriate record into EMP_SALARY table. Refer the following table for salary details.

Designation	Basic	DA	HRA	Deduction
Staff nurse	6000	2000	2000	2% of basic
Head nurse	8000	2500	3000	2% of basic
Technician	6000	2000	2000	2% of basic
Senior technician	9000	2500	3500	2.5% of basic
Junior attender	5000	1500	2000	2% of basic
Senior attender	6500	2000	2000	2% of basic

CODE:

```
use hospital;

CREATE TRIGGER emp_sal_update AFTER INSERT ON staff FOR EACH ROW

BEGIN

   IF NEW.Designation = 'Staff nurse' THEN          INSERT INTO emp_salary VALUES (NEW.Staff_Id, 6000, 6000, 2000, 2, (6000 + 6000 + 2000 - (0.02*6000)));

          ELSEIF New.Designation = 'Head nurse' THEN INSERT INTO emp_salary VALUES (NEW.Staff_Id, 8000, 2500, 3000, 2, (8000 + 2500 + 3000 - (0.2*8000)));

          ELSEIF New.Designation = 'Technician' THEN INSERT INTO emp_salary VALUES (NEW.Staff_Id, 6000, 2000, 2000, 2, (6000 + 2000 + 2000 - (0.2*6000)));

          ELSEIF New.Designation = 'Senior technician' THEN INSERT INTO emp_salary VALUES (NEW.Staff_Id, 9000, 2500, 3500, 2.5, (9000 + 2500 + 3500 - (0.25*9000)));

          ELSEIF New.Designation = 'Junior attender' THEN INSERT INTO emp_salary VALUES (NEW.Staff_Id, 5000, 1500, 2000, 2, (5000 + 1500 + 2000 - (0.2*5000)));
```

```
        ELSEIF New.Designation = 'Senior attender' THEN INSERT INTO emp_salary VALUES
(NEW.Staff_Id, 6500, 2000, 2000, 2, (6500 + 2000 + 2000 - (0.2*6500)));

  END IF;

END//

DELIMITER ;

INSERT INTO staff VALUES ('S0005', 'Throngan', 'nurse', 'Staff nurse', '2-Oct-1989', 7541097876,'44
Green View Road','D102' );
```

OUTPUT SCREENSHOT: