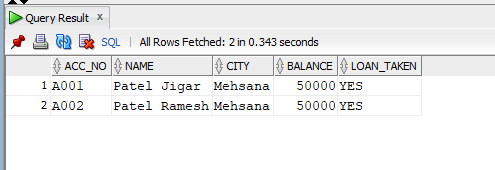
**PRACTICAL-4**

**1) Display the sum of balance of account holders who’s live in same city ‘Mehsana’ using group by clause.**

**SQL statement:**

select sum(balance) from account GROUP by city HAVING city='Mehsana';

**Output:**

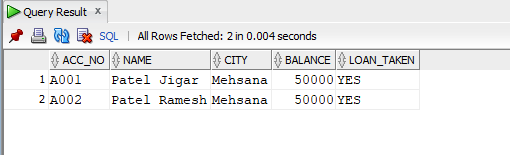
****

**2) Display the information about account where balance is less than total balance of all account holders.**

**SQL statement:**

select \* from account where balance< (SELECT AVG(balance) FROM account);

**Output:**

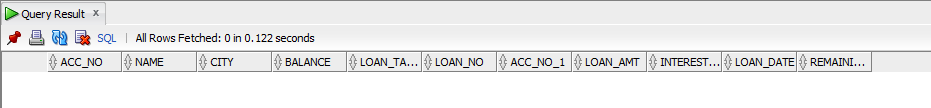
****

**3) Displays the information of account holders whose loan amount and balance both are same.**

**SQL statement:**

select \* from account join loan on account.acc\_no=loan.acc\_no where account.balance = loan.loan\_amt;

**Output:**

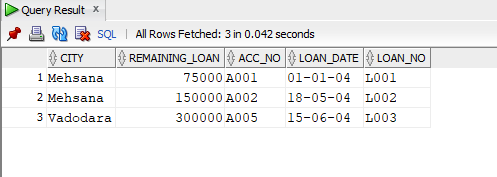
****

**4) Display the name of city, remaining loan amount, account, date of loan and loan number of account holders.**

**SQL statement:**

select a.city,l.remaining\_loan,a.acc\_no,l.loan\_date,l.loan\_no from account a,loan l where a.acc\_no=l.acc\_no;

**Output:**

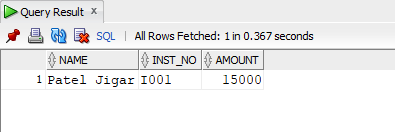
****

**5) Display name of account holder, installment number and installment amount Whose loan number is ‘L001’.**

**SQL statement:**

select ac.name,ins.inst\_no,ins.amount from account ac,loan ln,installment ins where ac.acc\_no=ln.acc\_no and ln.loan\_no=ins.loan\_no and ln.loan\_no='L001';

**Output:**

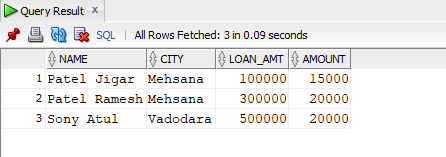
****

**6) Display name of account holder, city, loan amount and installment amount.**

**SQL statement:**

select ac.name,ac.city,ln.loan\_amt,ins.amount from account ac,loan ln,installment ins where ac.acc\_no = ln.acc\_no and ln.loan\_no=ins.loan\_no;

**Output:**

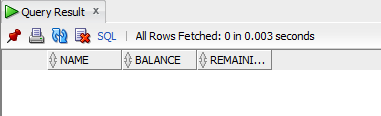
****

**7) Display the balance of account holders whose balance and remaining loan both are same.**

**SQL statement:**

select ac.name ,ac.balance,ln.remaining\_loan from account ac,loan ln where ac.acc\_no=ln.acc\_no and ac.balance=ln.remaining\_loan;

**Output:**

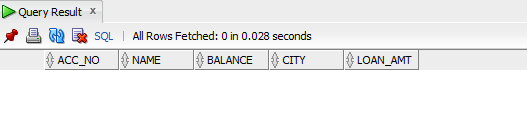
****

**8) List of all account holders’ information whose balance is same as loan amount.**

**SQL statement:**

select ac.acc\_no,ac.name,ac.balance,ac.city,ln.loan\_amt from account ac,loan ln where ac.acc\_no=ln.acc\_no and ac.balance=ln.loan\_amt;

**Output:**

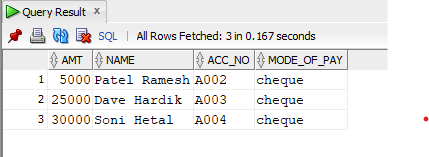
****

**9) Display the amount of transaction, name of account holders, account number and mode of payment whose mode of payment is ‘CHEQUE’.**

**SQL statement:**

select txn.amt,ac.name, ac.acc\_no,txn.mode\_of\_pay from account ac,transaction txn where ac.acc\_no=txn.acc\_no and txn.mode\_of\_pay='cheque';

**Output:**

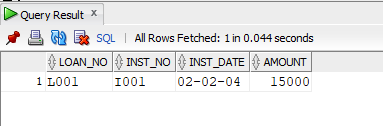
****

**10) List of installment information whose amount is less than average amount of transaction.**

**SQL statement:**

select \* from installment where amount < (select avg(amt) from transaction);

**Output:**

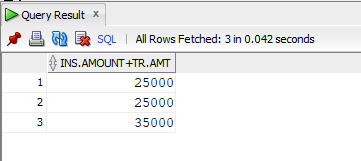
****

**11)** **Display the sum of installment amount and transaction amount.**

**SQL statement:**

select ins.amount+tr.amt from transaction tr,installment ins,loan ln where tr.acc\_no=ln.acc\_no and ins.loan\_no=ln.loan\_no;

**Output:**

****

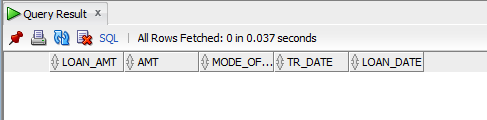
**12) Display loan amount, transaction amount and mode of payment where transaction date and loan taken date both are done in month of ‘MAY’.**

**SQL statement:**

select ln.loan\_amt,tr.amt,tr.mode\_of\_pay,tr.tr\_date,ln.loan\_date from transaction tr, loan ln where tr.acc\_no=ln.acc\_no

and to\_char(tr.tr\_date,'mon')='may' and to\_char(ln.loan\_date,'mon')='may';

**Output:**

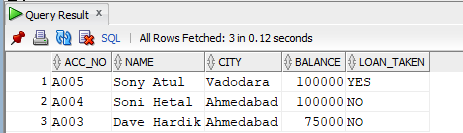
****

**13) Display the last three row of account table.**

**SQL statement:**

select \* from (select \* from account order by acc\_no desc) where rownum <=3;

**Output:**

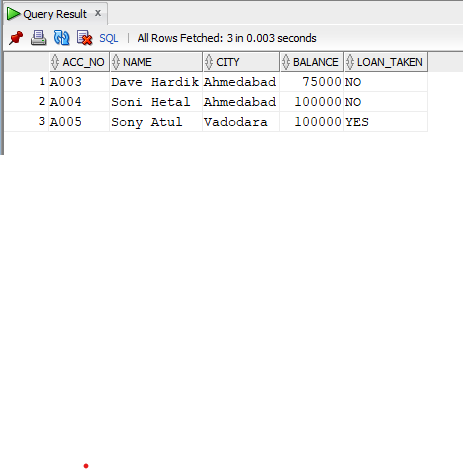
****

**14)** **Retrieve only rows 2 to 5 from account table.**

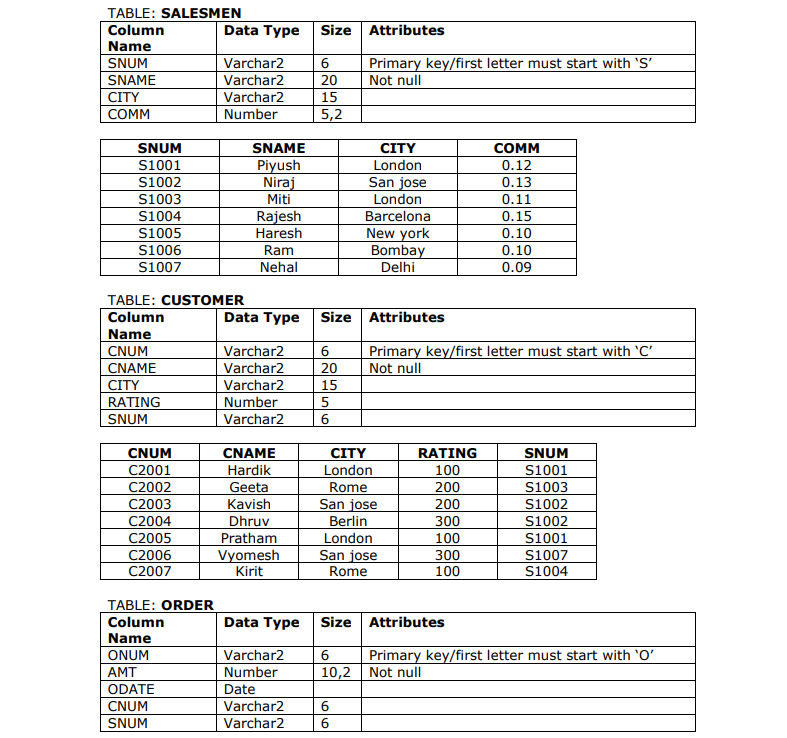
**SQL statement:**

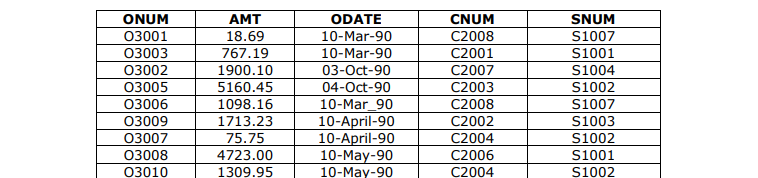
select \* from account where rownum<=5 minus select \* from account where rownum<=2;

**Output:**

****

**PRACTICAL-5**

****

****

****

**Salesman:**

create table salesmen (SNUM Varchar2(6) primary key , SNAME Varchar2(20) not null , CITY Varchar2(15) , COMM Number(5,2) , check (SNUM like 'S%') );

insert into salesmen values ('S1001' ,'Piyush' ,'London' ,0.12);

insert into salesmen values('S1002', 'Niraj ','San jose',0.13 );

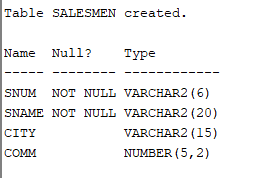
insert into salesmen values ('S1003' ,'Miti' ,'London' ,0.11 );

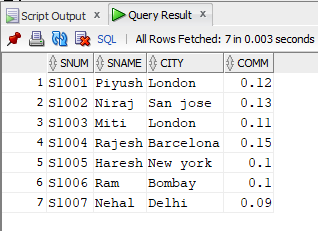
insert into salesmen values ('S1004' ,'Rajesh', 'Barcelona', 0.15 );

insert into salesmen values ('S1005' ,'Haresh' ,'New york' ,0.10 );

insert into salesmen values ('S1006' ,'Ram' ,'Bombay', 0.10 );

insert into salesmen values ('S1007' ,'Nehal', 'Delhi', 0.09 );





**customer:**

create table customer(CNUM Varchar2(6) primary key, CNAME Varchar2(20) not null , CITY Varchar2(15) , RATING Number(5), SNUM Varchar2(6) , check(CNUM like 'C%'));

insert into customer values('C2001', 'Hardik', 'London' , 100 ,'S1001' );

insert into customer values('C2002', 'Geeta', 'Rome' ,200 ,'S1003' );

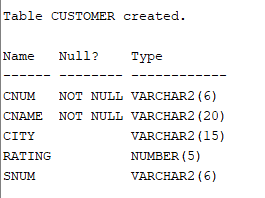
insert into customer values('C2003', 'Kavish', 'San jose' ,200, 'S1002' );

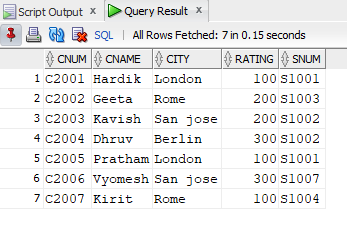
insert into customer values('C2004', 'Dhruv', 'Berlin' ,300 ,'S1002' );

insert into customer values('C2005', 'Pratham' ,'London', 100 ,'S1001' );

insert into customer values('C2006', 'Vyomesh' ,'San jose' ,300 ,'S1007' );

insert into customer values('C2007', 'Kirit' ,'Rome', 100, 'S1004' );





**order:**

create table orders(ONUM varchar2(6) primary key , AMT number(10,2) not null, ODATE date , CNUM varchar2(6) , SNUM varchar2(6), check(ONUM like 'O%'));

insert into orders values('O3001', 18.69 , '10-Mar-90', 'C2008', 'S1007');

insert into orders values('O3003', 767.19, '10-Mar-90', 'C2001' ,'S1001' );

insert into orders values('O3002', 1900.10 ,'03-Oct-90', 'C2007', 'S1004' );

insert into orders values('O3005', 5160.45, '04-Oct-90' ,'C2003' ,'S1002' );

insert into orders values('O3006', 1098.16 ,'10-Mar\_90', 'C2008', 'S1007' );

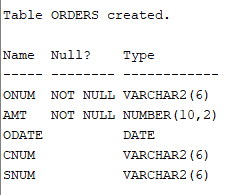
insert into orders values('O3009' ,1713.23 ,'10-April-90', 'C2002' ,'S1003');

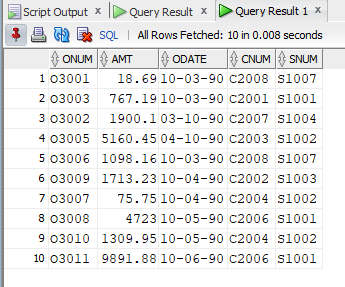
insert into orders values('O3007', 75.75 ,'10-April-90', 'C2004' ,'S1002');

insert into orders values('O3008' ,4723.00, '10-May-90', 'C2006' ,'S1001' );

insert into orders values('O3010' ,1309.95 ,'10-May-90', 'C2004' ,'S1002' );

insert into orders values('O3011' ,9891.88 ,'10-June-90', 'C2006' ,'S1001' );





**Perform following queries.**

**SELECT**

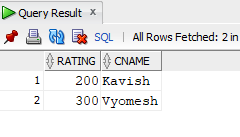
**1) Write a select command that produces the rating followed by the name of each customer in**

**SAN JOSE.**

**SQL statement:**

select rating,cname from customer where city='San jose';

**Output:**

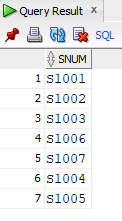


**2) Display SNUM values of all salesmen without any repeat.**

**SQL statement:**

select distinct snum from salesmen;

**Output:**



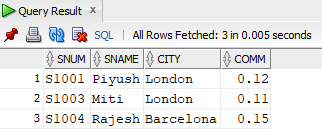
**SPECIAL OPERATORS:**

**3) Display all salesmen that were located in either BARCELONA or LONDON(use IN keyword).**

**SQL statement:**

select \*from salesmen where city in('Barcelona','London');

**Output:**

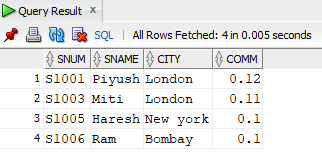


**4) Display all salesmen with commission between 0.10 and 0.12.**

**SQL statement:**

select \*from salesmen where comm between 0.10 and 0.12;

**Output:**



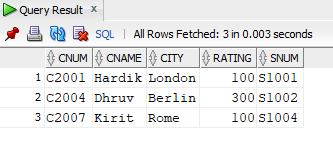
**LIKE OPERATORS:**

**5) List all the customers whose name’s third latter is ‘R’.**

**SQL statement:**

select \*from customer where cname like '\_\_r%';

**Output:**

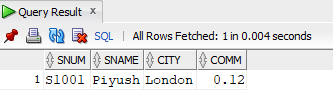


**6) List all salesmen whose sname start with letter ‘P’ and end letter is ‘H’.**

**SQL statement:**

select \*from salesmen where SNAME like 'P%h';

**Output:**



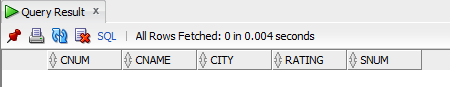
**NULL OPERATORS:**

**7) Find all records in customer table with NULL values in the city column.**

**SQL statement:**

select \*from customer where city=null;

**Output:**

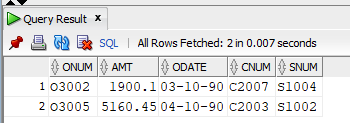


**8) Write a two queries that will produce all orders taken on October 3rd or 4th ,1990 (Use BETWEEN operator )**

**SQL statement:**

select \*from orders where odate between '03-Oct-90' and '04-Oct-90';

**Output:**

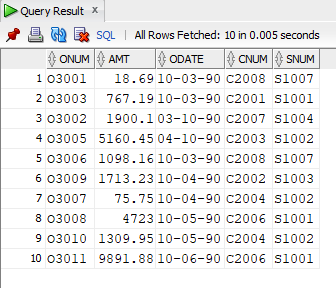


**9) Write a query that selects all orders without ZEROS or NULLS in amt field.**

**SQL statement:**

select \*from orders where amt is not null and amt!=0;

**Output:**



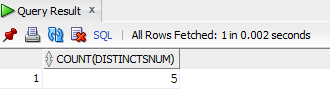
**FUNCTIONS:**

**10) To count the numbers of salesmen without duplication in the orders tables.**

**SQL statement:**

select count(distinct snum) from orders;

**Output:**

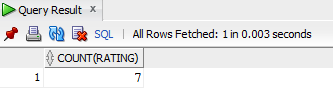


**11)** **Count the rating of customers (with NULL and without NULL).**

**SQL statement:**

select count(rating) from customer

**Output:**

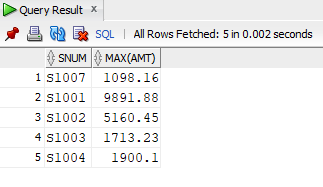


**12) Find the largest order taken by each salesperson.(hint: use group by)**

**SQL statement:**

select snum,max(amt) from orders group by snum;

**Output:**

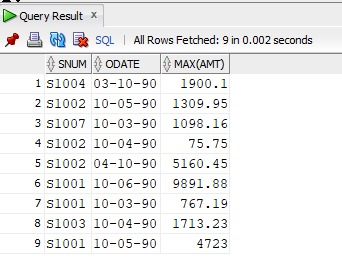


**13) Find the largest order taken by each salesperson on each date.**

**SQL statement:**

select snum,odate,max(amt) from orders group by snum,odate;

**Output:**

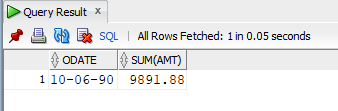


**14)** **Find out which day had the higher total amount ordered.**

**SQL statement:**

select odate,sum(amt) from orders group by odate having sum(amt)= (select max(sum(amt)) from orders GROUP by odate);

**Output:**

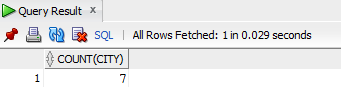


**15)** **Write a query that counts the number of different non-NULL city in the customer table.**

**SQL statement:**

select count(city) from customer where city is not null;

**Output:**

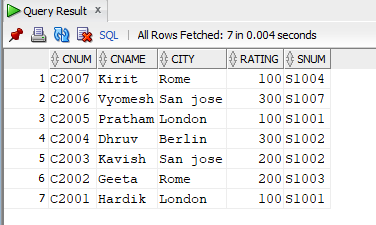


**16)** **Display all the information in descending orders(use column CNUM).**

**SQL statement:**

select \*from customer order by cnum desc;

**Output:**

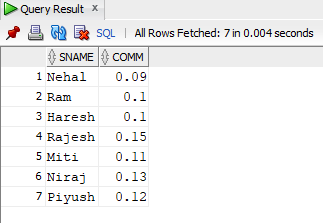


**17)** **Display sname and comm. From salesmen in descending order(in place of column name use column number).**

**SQL statement:**

select sname,comm from salesmen order by snum desc;

**Output:**

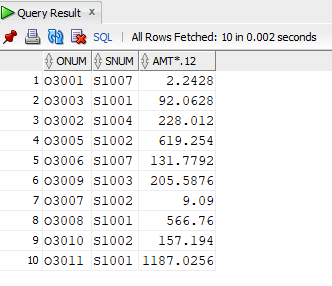


**18)** **Assume each salesperson has a 0.12 commission. Write a query on the orders table that will produce the order number,the salesperson number and the amount of the salesperson’s commission for that order.**

**SQL statement:**

select onum,snum,amt\*.12 from orders;

**Output:**



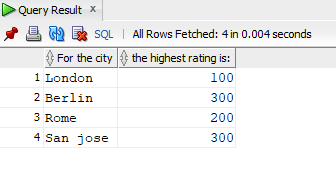
**19)** **Write a query on the customers table that will find the highest rating in each city. Put the output in this form.**

**For the city (city) , the highest rating is: (rating).**

**SQL statement:**

select city "For the city",max(rating) "the highest rating is:" from customer group by city;

**Output:**

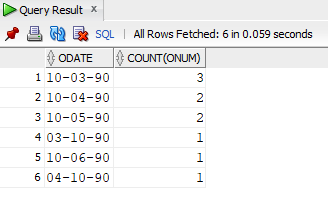


**20)** Write a query that totals the orders for each day and places the results in descending order.

**SQL statement:**

select odate,count(onum) from orders group by odate order by count(onum) desc;

**Output:**



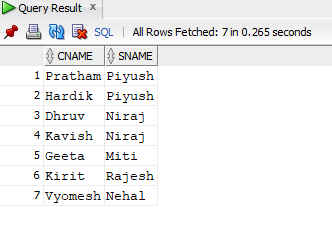
**JOIN:**

**21) Show the names of all customers matched with the salesmen serving them**

**SQL statement:**

select customer.cname,salesmen.sname from customer,salesmen where customer.snum=salesmen.snum;

**Output:**

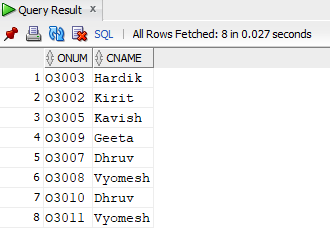


**22)** **Write a query that lists each order number followed by the name of the customer who made the order.**

**SQL statement:**

select orders.onum,customer.cname from customer,orders where customer.cnum=orders.cnum;

**Output:**

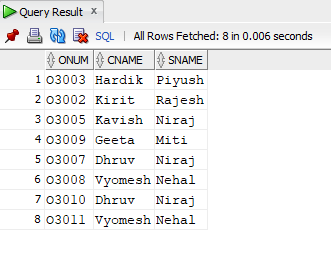


**23)** **Write a query that gives the names of both the salesperson and the customer for each order after the order number.**

**SQL statement:**

select orders.onum,customer.cname,salesmen.sname from customer,salesmen,orders where customer.cnum=orders.cnum and customer.snum=salesmen.snum;

**Output:**

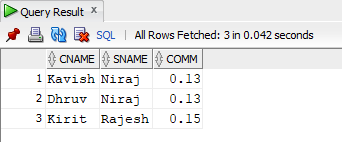


**24)** **Write a query that produces all customers serviced by salesmen with a commission above 0.12. Output the customer’s name, the salesperson’s name and the salesperson’s rate of commission.**

**SQL statement:**

select customer.cname,salesmen.sname,salesmen.comm from customer,salesmen where customer.snum=salesmen.snum and salesmen.comm>.12 ;

**Output:**

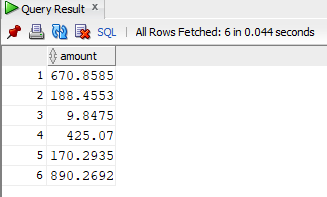


**25)** **Write a query that calculates the amount of the salesperson’s commission on each order by a customer with a rating above 100.**

**SQL statement:**

select orders.amt\*salesmen.comm "amount" from customer,salesmen,orders where customer.snum=salesmen.snum and customer.cnum=orders.cnum and rating>100;

**Output:**



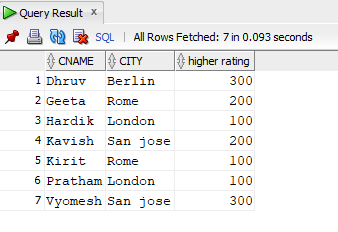
**OTHERS:**

**26)** **Create a union of two queries that shows the names,cities and ratings of all customers. Those with rating of >=200 should display ‘HIGH RATING’ and those with**

**SQL statement:**

select cname,city,rating "higher rating" from customer where rating>=200 union select cname,city,rating "lower rating" from customer where rating<200;

**Output:**



**27)** **Find all customers with orders on 3rd october 1990 using correlate sub query.**

**SQL statement:**

select cname from customer where cnum = (select cnum from orders where odate='03-Oct-90');

**Output:**

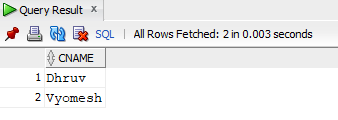


**28)** **Find all customers having rating greater than any customer in ‘ROME’.**

**SQL statement:**

select cname from customer where rating > (select max(rating) from customer where city='Rome');

**Output:**

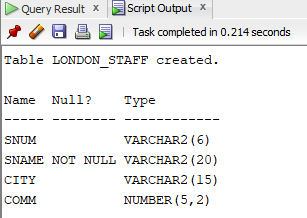


**29)** **Create another table London\_staff having same structure as salesmen table.**

**SQL statement:**

create table London\_staff as select \* from salesmen where 1=2;

**Output:**



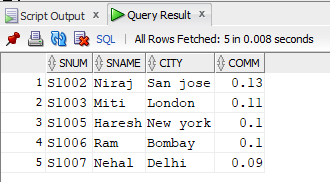
**30)** **Delete all salesmen who have at least one customer with a rating of 100 from salesmen table.**

**SQL statement:**

delete from salesmen where snum in (select snum from customer where rating=100);

**Output:**





**PRACTICAL-6**

**1) Write a pl/sql code for calculator. Accept two values from user and accept any operator like ‘+’,’-‘,’\*’,’/’.**

**PL / SQL statement:**

DECLARE

p number:=&p;

q number:=&q;

opr varchar2(1):='&operator';

BEGIN

case(opr)

when '+' then dbms\_output.put\_line('answer is:'||(p+q));

when '-' then dbms\_output.put\_line('answer is:'||(p-q));

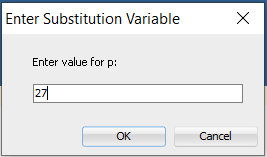
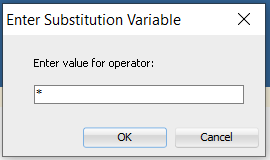
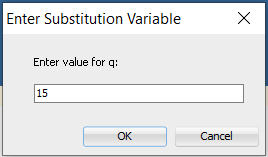
when '\*' then dbms\_output.put\_line('answer is:'||(p\*q));

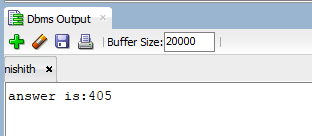
when '/' then dbms\_output.put\_line('answer is:'||(p/q));

end case;

end;

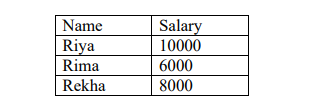
**Output:**

****

****

**2) Write a pl/sql code for add 50% salary of ‘Riya’ and 70% salary of ‘Rima’ to salary of ‘Rekha’.**

**Table: Employee**

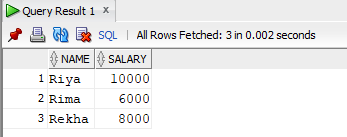
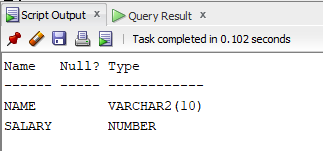
****

create table employees (name varchar2(10), salary number);

insert into employees values ('Riya',10000);

insert into employees values ('Rima',6000);

insert into employees values ('Rekha',8000);

****

**PL / SQL statement:**

DECLARE

a employees.salary%type;

b employees.salary%type;

BEGIN

SELECT salary into a from employees where name='Riya';

SELECT salary into b from employees where name='Rima';

update employees set salary=salary+a+0.5+b\*0.7 where name='Rekha';

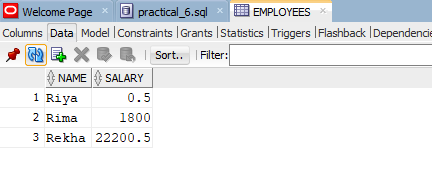
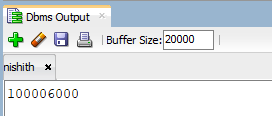
update employees set salary=salary-a+0.5 where name='Riya';

update employees set salary=salary-b\*0.7 where name='Rima';

dbms\_output.put\_line(a||b);

end;

**Output:**

****

**3) Write a pl/sql code to give maximum salary of Rima or Rekha to Riya.**

**PL / SQL statement:**

DECLARE

a employees.salary%type;

b employees.salary%type;

BEGIN

select salary into a from employees where name='Rima';

select salary into b from employees where name='Rekha';

if a>b then

update employees set salary=a where name='Riya';

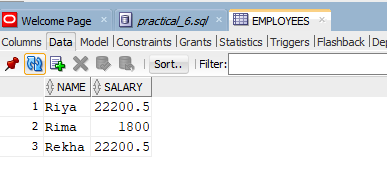
else

update employees set salary=b where name='Riya';

end if;

end;

**Output:**

****

**4) Write a pl/sql code that will accept emp\_no and debit amount from user. If Balance is below 500/- whenever you debit the amount then give error message. “Balance below 500/-“otherwise update the table.**

**PL / SQL statement:**

DECLARE

a employees.salary%type;

b employees.salary%type;

c employees.salary%type;

d employees.salary%type;

empno number:=&empno;

debit number:=&debit;

BEGIN

select salary into a from employee where emp\_no=1001;

select salary into b from employee where emp\_no=1002;

select salary into c from employee where emp\_no=1003;

select salary into d from employee where emp\_no=1004;

if empno=1001 then

if (a-500-debit) >= 0 then

update employee set salary =salary -debit where emp\_no=1001;

else

dbms\_output.put\_line('low balance to debit');

end if;

end if;

if empno=1002 then

if (b-500-debit) >= 0 then

update employee set salary =salary -debit where emp\_no=1002;

else

dbms\_output.put\_line('low balance to debit');

end if;

end if;

if empno=1003 then

if (c-500-debit) >= 0 then

update employee set salary =salary -debit where emp\_no=1003;

else

dbms\_output.put\_line('low balance to debit');

end if;

end if;

if empno=1004 then

if (d-500-debit) >= 0 then

update employee set salary =salary -debit where emp\_no=1004;

else

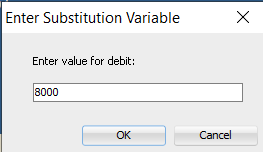
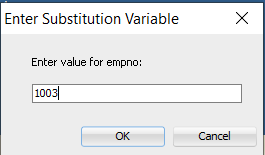
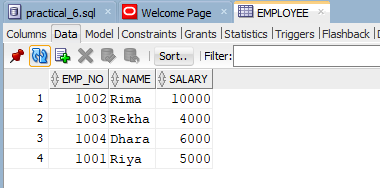
dbms\_output.put\_line('low balance to debit');

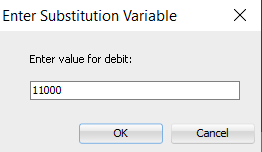
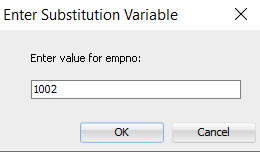
end if;

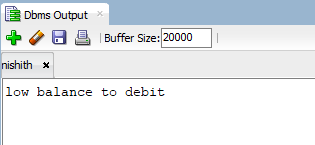
end if;

end;

**Output:**

****





**5) Write a pl/sql block for Factorial (use a) If and Goto statement b) for loop).**

**PL / SQL statement:**

**a)**

declare

a number ;

fact number:=1;

begin

a:=&a;

<<repeat\_loop>>

fact:=fact\*a;

a:=a-1;

if a>0 then

goto repeat\_loop;

end if;

dbms\_output.put\_line('fact(goto): '||fact);

end;

**b)**

declare

a number(2);

fact number:=1;

t number(2);

BEGIN

a:=&a;

for t in 1..a loop

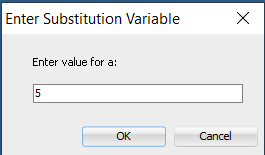
fact:=fact\*t;

end loop;

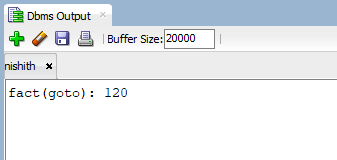
dbms\_output.put\_line('fact: '||fact);

end;

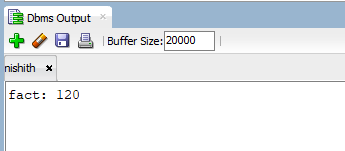
**Output:**

****

**a)**

****

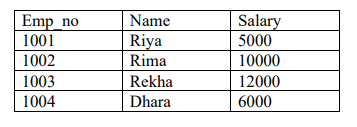
**b)**

****

**6) Write pl/sql code for following. If the salary of employee no 1001 is less than 6000 then change the salary to 6000. The salary change is to be recorded in the old\_salary table with date on which salary changed.**

**Table: old\_salary(emp\_no,old\_salary,change\_date)**

**Table: Salary**

****

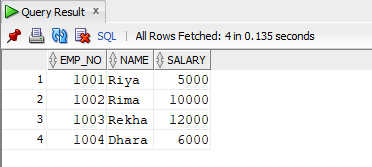
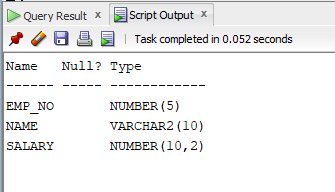
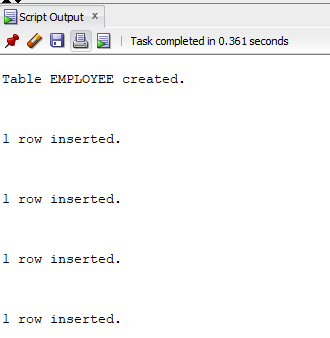
create table employee(emp\_no number(5),name varchar2(10),salary number(10,2));

insert into employee values(1001,'Riya',5000);

insert into employee values(1002,'Rima',10000);

insert into employee values(1003,'Rekha',12000);

insert into employee values(1004,'Dhara',6000);

****

**PL / SQL statement:**

create table old\_salary(emp\_no,old\_sal,change\_date) as select emp\_no,salary,sysdate from employee where emp\_no=1001;

declare

a employee.salary%type;

begin

select salary into a from employee where emp\_no=1001;

if a<6000 then

update old\_salary set old\_sal=(select salary from employee where emp\_no=1001) where emp\_no=1001;

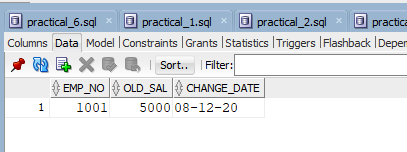
update employee set salary = 6000 where emp\_no=1001;

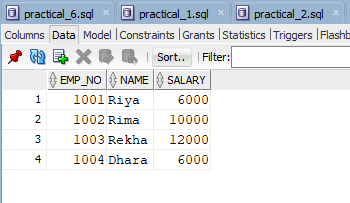
update old\_salary set change\_date=sysdate where emp\_no=1001;

end if;

end;

**Output:**

****

****

**PRACTICAL-7**

**1) Analyze output of following scripts:**

**PL / SQL statement:**

A. Begin

Insert into new\_salary values ('6', '6000');

Update new\_salary set salary=4000 where id=4; Commit; Rollback;

End; /

B. Begin

Insert into new\_salary values (6, 6000);

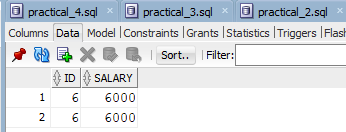
Savepoint first;

Update new\_salary set salary=50000 where id=5;

Rollback to savepoint first;

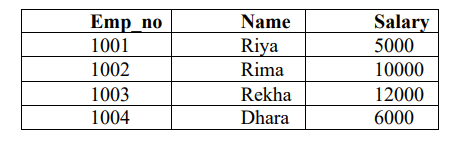
End; /

**Output:**

****

**2)** . **Write pl/sql blocks that first insert the new record into table salary. Update the salary of employee name ‘Riya’ and ‘Rima’ by 7000/- then check the total salary which does not exceed 20000/-.if it happens then UNDO the update and give the error message “TOTAL SALARY EXCEED 20000”.**

**Table:Salary**

****

**PL / SQL statement:**

declare

sal emp\_salary.salary%type;

begin

insert into emp\_salary values(1005,'Richa',8000);

savepoint try;

update emp\_Salary set salary=11000 where name in ('Riya','Rima');

select sum(salary) into sal from emp\_salary where name in ('Riya','Rima');

if sal>20000 then

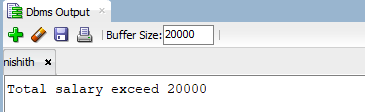
rollback to savepoint try;

dbms\_output.put\_line('Total salary exceed 20000');

end if;

end;

**Output:**

****

**3) Write a pl/sql to give 5% raise to every employee.**

**Use Table Salary.**

**PL / SQL statement:**

declare

a number;

b emp\_salary.name%type;

c number;

cursor my\_cursor is select \*from emp\_salary;

begin

update emp\_salary set salary=1.05\*salary;

open my\_cursor;

loop

fetch my\_cursor into a,b,c;

exit when my\_cursor%notfound;

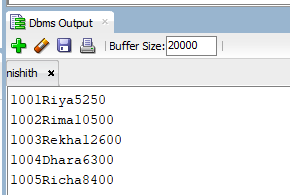
dbms\_output.put\_line(a||b||c);

end loop;

close my\_cursor;

end;

**Output:**

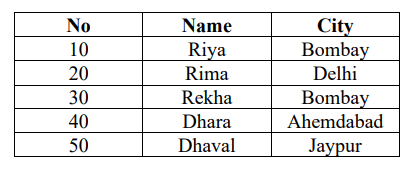
****

**4) Write a pl/sql block that find the table empmaster and stores the data into another table CITYLIST(no,name) where CITY=’Bombay’ . Create this pl/sql using following two:**

**Use pl/sql data type %TYPE**

**Use pl/sql data %ROWTYPE.**

**Table: Empmaster**

****

**PL / SQL statement:**

DECLARE

cursor cursor1 is select \* from empmaster where city = 'Bombay';

my\_row cursor1%rowtype;

begin

open cursor1;

loop

fetch cursor1 into my\_row;

exit when cursor1%notfound;

dbms\_output.put\_line(my\_row.no);

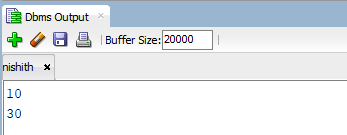
insert into citylist values(my\_row.no,my\_row.name);

end loop;

close cursor1;

end;

**Output:**

****

**5) Write a pl/sql block that update SALARY of employee of a specified company Using parameter in Cursor.**

**Table : Temp**

**PL / SQL statement:**

declare

cursor cursor1(comp varchar2) is select \*from temp where company=comp;

c varchar2(20);

my\_row cursor1%rowtype;

begin

c:=&c;

open cursor1(c);

loop

fetch cursor1 into my\_row;

exit when cursor1%notfound;

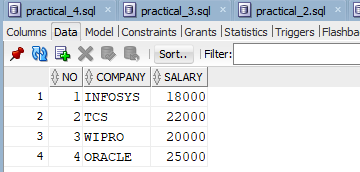
update temp set salary = 5000 where company=my\_row.company;

end loop;

close cursor1;

end;

**Output:**

****

**6)**  **Display Only NO,NAME of First three row . Use table Empmaster ( Use FOR LOOP)**

**PL / SQL statement:**

declare

cursor cursor1 is select \* from empmaster;

my\_row cursor1%rowtype;

t number;

begin

open cursor1;

for t in 1..3 loop

fetch cursor1 into my\_row;

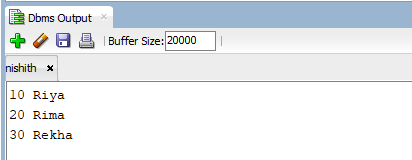
dbms\_output.put\_line(my\_row.no||' '||my\_row.name);

end loop;

close cursor1;

end;

**Output:**



**PRACTICAL-8**

**1) Create a Function which will work as addition. e.g select ADDITION(10,10) from dual;**

**PL / SQL statement:**

create or replace function adding(x in number,y in number)

return number

is

z number;

begin

z:=x+y;

return z;

end;

declare

t number;

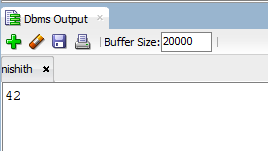
begin

t:=adding(15,27);

dbms\_output.put\_line(t);

end;

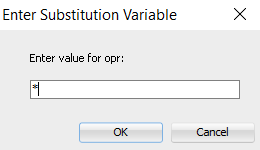
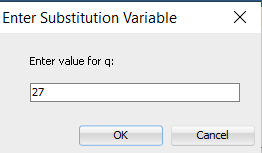
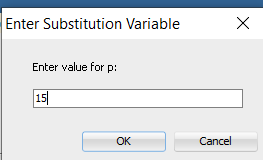
**Output:**

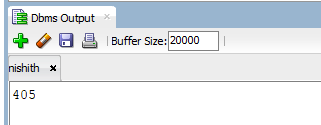
****

**2)** **Create a Function that is Work as Calculator. Use operator like ‘+’,’-’,’\*’,’/’.**

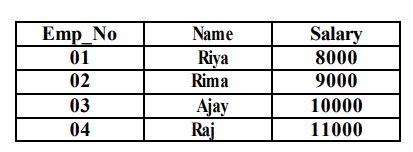
**PL / SQL statement:**

**Output:**

****

****

**3)** **Create a Procedure that update the salary of employee Using following formula: Sal = Sal + (Sal\* Percentage/100). Pass two parameter values: Name & Percentage. Table : Salary**

****

**PL / SQL statement:**

create or replace procedure changes(nm in varchar2,percentage in number)

is

begin

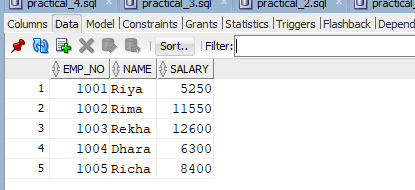
update employee set salary=salary+(salary\*percentage/100) where

name=nm;

end;

execute changes('Rima',10);

**Output:**

****

**4) When you perform update or delete operation on table company1, it stores the updated or deleted values in table company2. Also show the operation ‘update’ and ‘delete’ and username.**

**Table: Company1 (no, name) Company2 (no, name, operation)**

**PL / SQL statement:**

create or replace trigger t\_change

after update or delete

on company1

for each row

begin

if updating then

insert into company2 values(:old.no,:old.name,'update',user);

dbms\_output.put\_line('update opertion');

end if;

if deleting then

insert into company2 values(:old.no,:old.name,'delete',user);

dbms\_output.put\_line('delete opertion');

end if;

end;

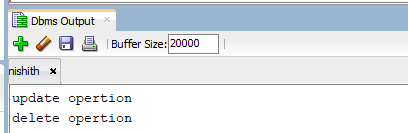
begin

update company1 set name='hello' where no=102;

delete from company1 where NO=106;

end;

**Output:**

****

**5)** **Write a pl/sql code to retrieve the data from student table whose sr.no is 10it48. If data not found then show the message “data not found“ (Use EXCEPTION in pl/sql code)**

**Table: STUDENT (sr\_no, name, address)**

**PL / SQL statement:**

declare

c\_id student1.sr\_no%type;

c\_name student1.name%type;

c\_addr student1.address%type;

begin

select name,address into c\_name,c\_addr from student1 where sr\_no='20it20';

exception

when no\_data\_found then

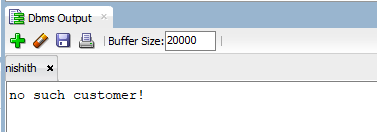
dbms\_output.put\_line('no such customer!');

when others then

dbms\_output.put\_line('error!');

end;

**Output:**

****