**CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY**

**FACULTY OF TECHNOLOGY & ENGINEERING**

U & P U. Patel Department of Computer Engineering

**Subject Name:** Database Management System **Semester :** IVth

**Subject Code:** CE246 **Academic Year :** Dec-May

Practical 1

**Aim : To study DDL-create and DML-insert commands.**

Theory :

1. **Create table** : We can create table using the following query :

create table table\_name(atr1 datatype(size),atr2 datatype(size),…);

This query create a table with name table\_name and its attributes are atr1, atr2, etc… .

1. **Describe table** : To describe the table we can write the following query :

desc table\_name; OR describe table\_name;

It shows the attributes names and their datatypes.

1. **Insert records into table** : For example if a table is of deposit and its attributes are accno, cname, bname, amount and adate then we can insert records in 3 types :
2. insert all data : We have write the following query for insert all data –

insert into deposit(accno, bname, cname, amount, adate)

values(‘100’,’abcd’,’xyz’,5000,’2-jan-1942’);

1. insert specific data : For this the query is-

insert into deposit(accno, bname, amount)

values(‘100’, ‘xyz’, 5000);

1. insert from the user after executing the query :

insert into deposit(accno, cname, bname, amount, adate)

values(‘&accno’, ‘&cname’, ‘&bname’, amount, ‘&adate’);

1. **Select query** : This query is used to show the data of table.
2. To show all the data of table : select \* from table\_name;

This query shows all the data of table.

1. For specific data : select accno , cname from deposit;

This query shows account no and customer name from the deposit table.

1. Use of condition : We can write condition using ‘where’ clause in the select query.

For example if we want data from deposit table which are from the branch VRCE so we can write the following query –

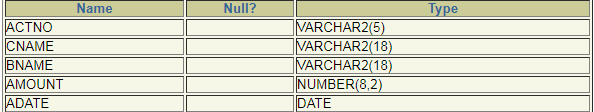
select \* from deposit where bname = ‘ VRCE’ ;

Now if we want the records from deposit table in which customers were joined between two dates. For this the query is –

select \* from deposit where adate between ’01-JUN-1997’ and ’01-JUN-1998’;

CREATE TABLE DEPOSIT (ACTNO VARCHAR2(5) ,CNAME VARCHAR2(18) , BNAME VARCHAR2(18) , AMOUNT NUMBER(8,2) ,ADATE DATE);

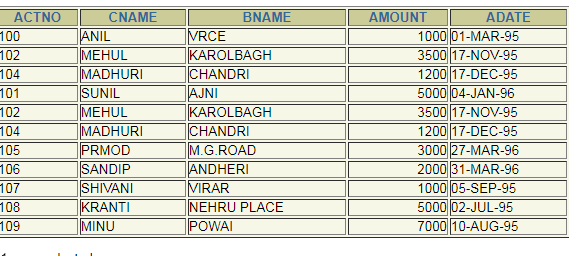
desc DEPOSIT;



insert into DEPOSIT values(100,'ANIL','VRCE',1000.00,1-MAR-95);

insert into DEPOSIT values(‘&ACTNO’,’&CNAME’,’&BNAME’,&AMOUNT,’&ADATE’);

select \* from DEPOSIT;



CREATE TABLE BRANCH(BNAME VARCHAR2(18),CITY VARCHAR2(18));

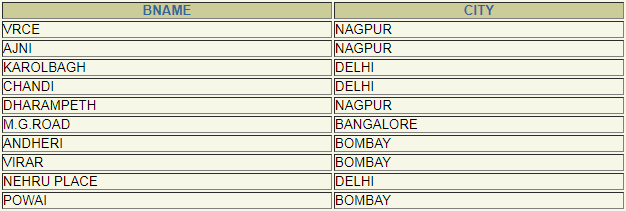
desc BRANCH;

C:\Users\Administrator\Desktop\desc(BRANCH).PNG

insert into BRANCH values('VRCE','NAGPUR');

insert into BRANCH values(‘&BNAME’,’&CITY’);

SELECT \* FROM BRANCH;



CREATE TABLE CUSTOMERS(CNAME VARCHAR2(19) ,CITY VARCHAR2(18));

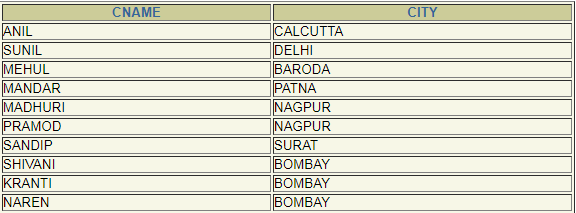
desc Customers;

Description: C:\Users\Administrator\Desktop\dbms customers(desc).PNG

insert into CUSTOMERS values(' ANIL',' CALCUTTA');

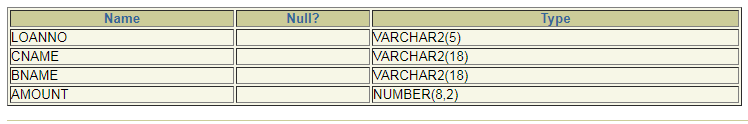
insert into CUSTOMERS values(‘&CNAME’,’&CITY’);

select \* from CUSTOMERS;



CREATE TABLE BORROW(LOANNO VARCHAR2(5), CNAME VARCHAR2(18), BNAME VARCHAR2(18), AMOUNT NUMBER (8,2));

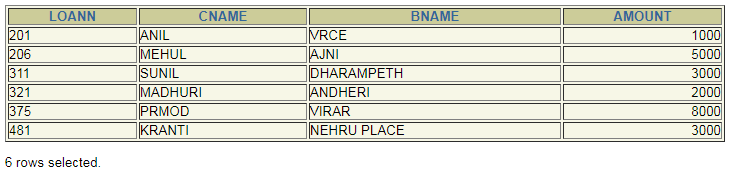
desc BORROW;



insert into BORROW values(201,'ANIL','VRCE',1000.00);

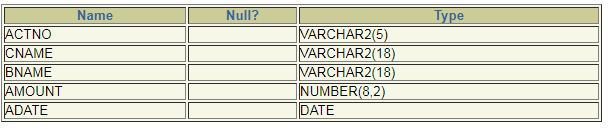
insert into BORROW values(‘&LOANN’,’&CNAME’,’&BNAME’,&AMOUNT);

select \* from BORROW;

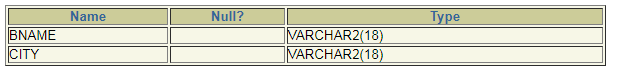


1. Describe deposit, branch.

desc deposit;

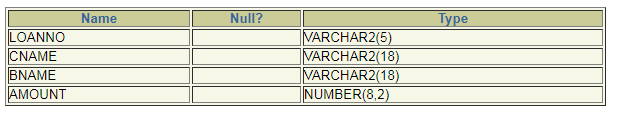


desc branch;

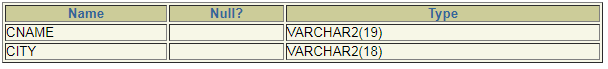


1. Describe borrow, customers.

desc borrow;

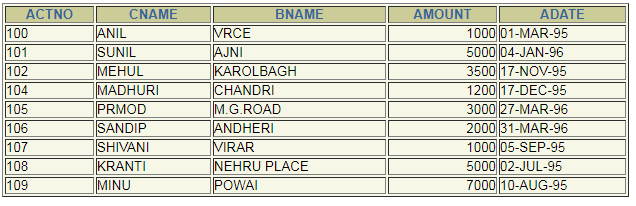


desc customers;



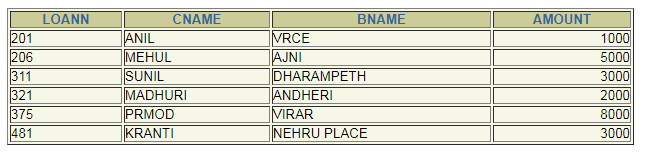
1. List all data from table DEPOSIT.

select \* from deposit;



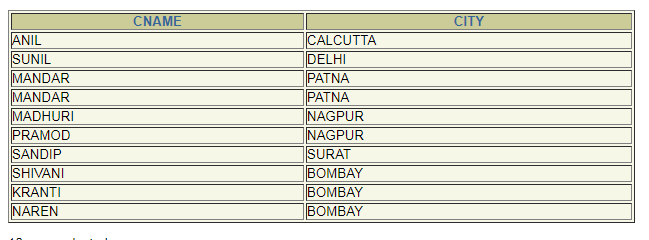
1. List all data from table BORROW.

select \* from borrow;



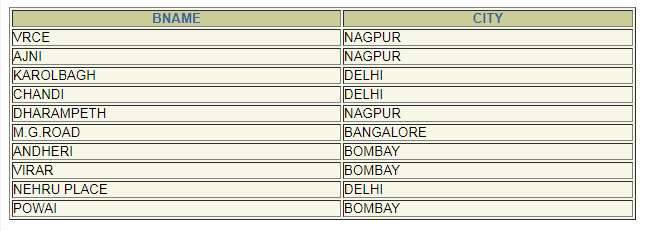
1. List all data from table CUSTOMERS.

select \* from customers;



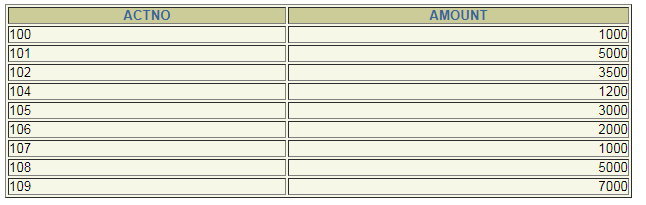
1. List all data from table BRANCH.

select \* from branch;



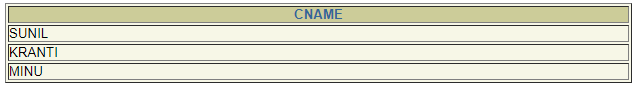
1. Give account no and amount of depositors.

select actno,amount from deposit



1. Give name of depositors having amount greater than 4000.

select cname from deposit where amount>4000



1. Give name of customers who opened account after date '1-12-96'.

select cname from deposit where adate>'1-dec-96'



1. Give name of city where branch karolbagh is located.

select bname from branch where city = 'karolbagh'



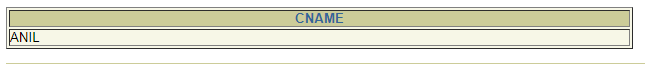
1. Give account no and amount of customer having account opened between date 1-12-96 and 1-6-96.

select actno,amount from deposit where adate between '1-jun-96' and '1-dec-96';



1. Give names of depositors having account at VRCE.

select cname from deposit where bname='VRCE'



PRACTICAL-2

AIM : **Create the below given table and insert the data accordingly**.

Theory :

**BETWEEN…AND** operators in SQL are used to select in-between values from the given range/ values.

e.g SELECT column\_name1, column\_name2, etc  
FROM table\_name  
WHERE  column\_name1 **BETWEEN** value1 **AND** value2;

The **CONCAT()** function concatenates two or more expressions together.

Syntax : CONCAT(expression1, expression2, expression3,...);

**IS NOT NULL**

It is used to print non null values of the table.

e.g SELECT LastName, FirstName, Address FROM Persons  
WHERE Address IS NOT NULL;

Creating tables,

CREATE TABLE JOB(JOB\_ID VARCHAR2(15), JOB\_TITLE VARCHAR2(30), MIN\_SAL NUMBER(7,2), MAX\_SAL NUMBER(7,2));

CREATE TABLE EMPLOYEE(EMO\_NO NUMBER(3), EMP\_NAME VARCHAR2(30), EMP\_SAL NUMBER(8,2), EMP\_COMM NUMBER(6,1), DEPT\_NO NUMBER(3);

CREATE TABLE DEPOSIT1(A\_NO VARCHAR2(5), CNAME VARCHAR2(15), BNAME VARCHAR2(10), AMOUNT NUMBER(7,2),A\_DATE DATE);

CREATE TABLE BORROW(LOANNO VARCHAR2(5), CNAME VARCHAR2(15), BNAME VARCHAR2(10), AMOUNT VARCHAR2(7,2));

TABLE JOB CREATED.

TABLE EMPLOYEE CREATED.

TABLE DEPOSIT1 CREATED.

TABLE BORROW CREATED.

Entering values in table.

INSERT INTO JOB VALUES(‘&JOB\_ID’,’&JOB\_TITLE’,’&MIN\_SAL’,’&MAX\_SAL’);

//entities are added to the table as per the question.

INSERT INTO EMPLOYEE VALUES(&EMP\_NO,’&EMP\_NAME’, &EMP\_SAL, &EMP\_COMM, &DEPT\_NO);

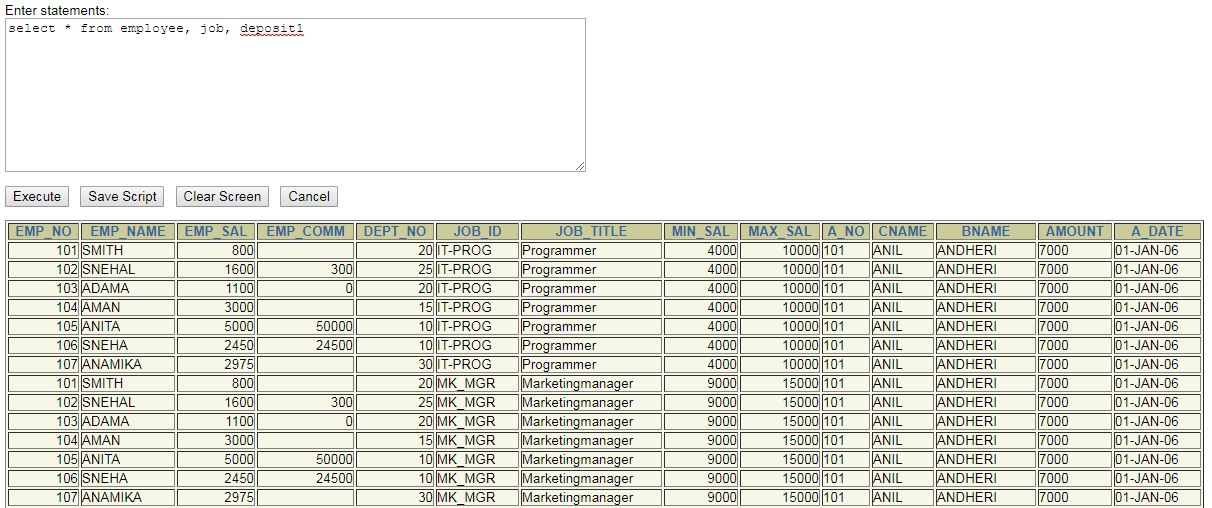
//entities are added to the table as per the question.

INSERT INTO DEPOSIT1 VALUES(‘&A\_NO’,’&CNAME’,’&BNAME’,&AMOUNT,’&A\_DATE’);

//entities are added to the table as per the question.

Perform following queries

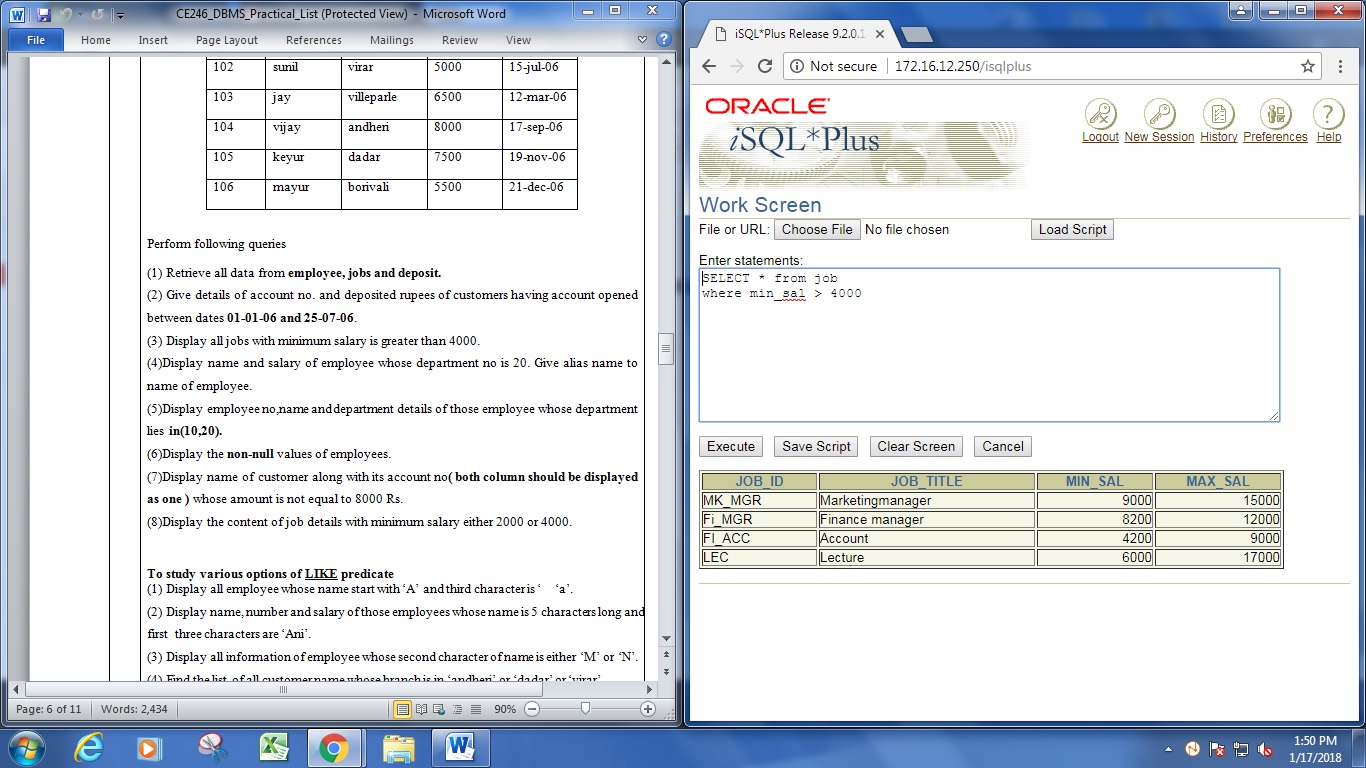
(1) Retrieve all data from **employee, jobs and deposit.**



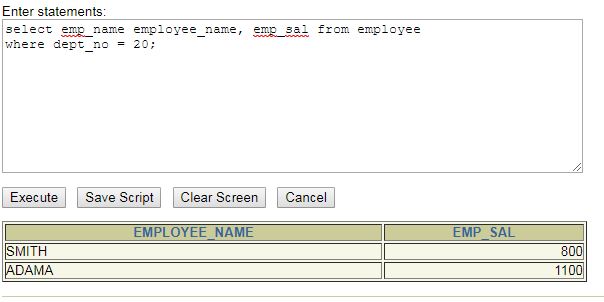
(2) Give details of account no. and deposited rupees of customers having account opened between dates **01-01-06 and 25-07-06**.



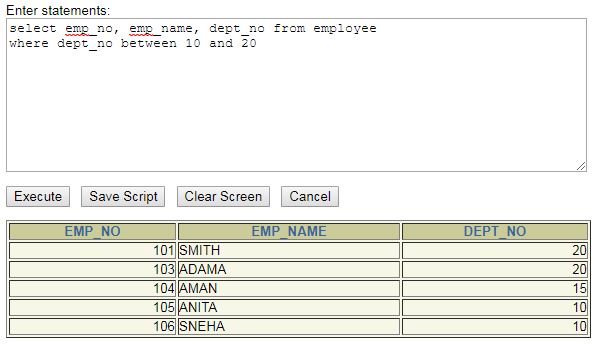
(3) Display all jobs with minimum salary is greater than 4000.



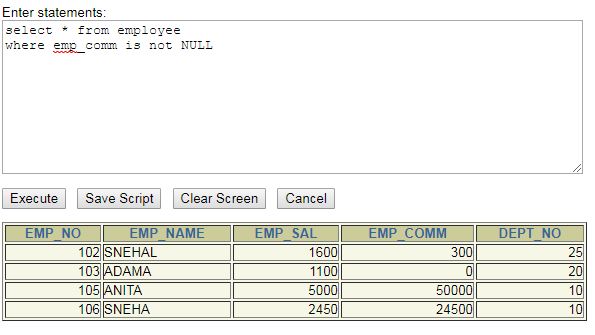
(4)Display name and salary of employee whose department no is 20. Give alias name to name of employee.

 employee.

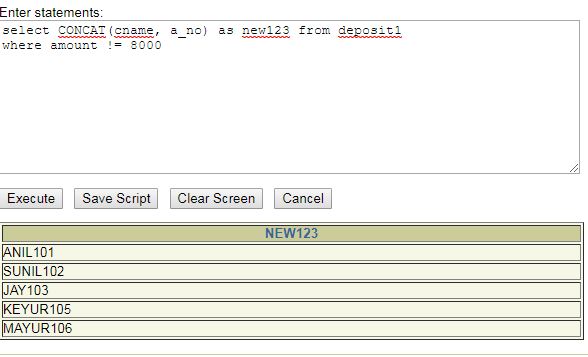
(5)Display employee no,name and department details of those employee whose department lies **in(10,20).**

****

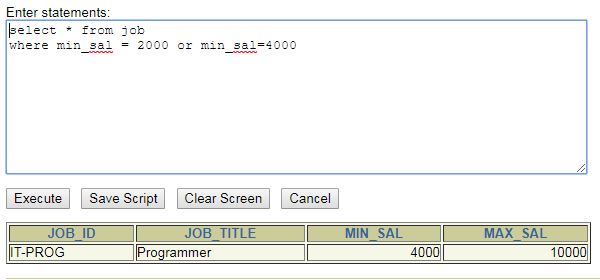
(6)Display the **non-null** values of employees.



(7)Display name of customer along with its account no**( both column should be** **displayed as one )** whose amount is not equal to 8000 Rs.

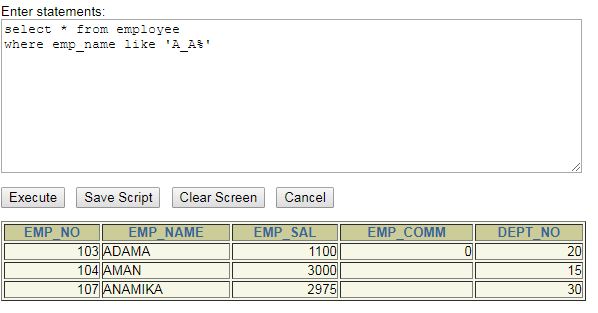


(8)Display the content of job details with minimum salary either 2000 or 4000.

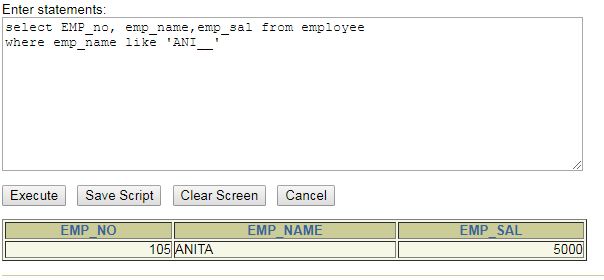


## LIKE Predicate

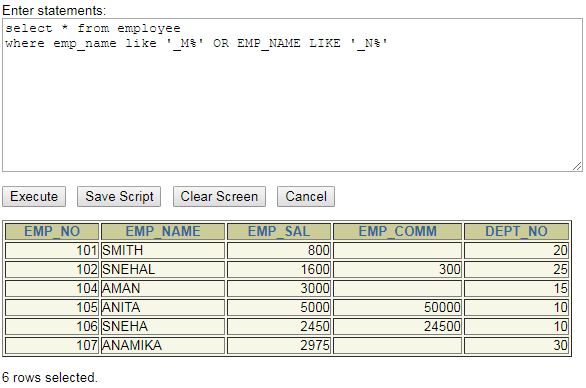
1. Display all employee whose name start with ‘A’ and third character is ‘ ‘a’.



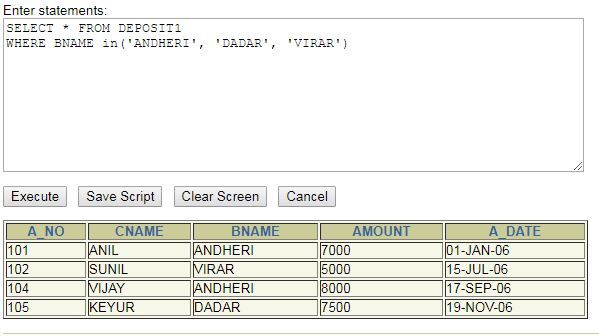
(2) Display name, number and salary of those employees whose name is 5 characters long and first three characters are ‘Ani’.



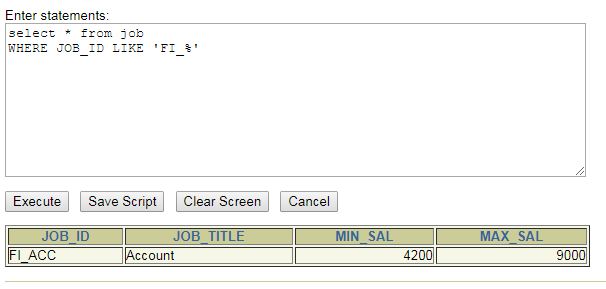
(3) Display all information of employee whose second character of name is either ‘M’ or ‘N’.



(4) Find the list of all customer name whose branch is in ‘andheri’ or ‘dadar’ or ‘virar’.



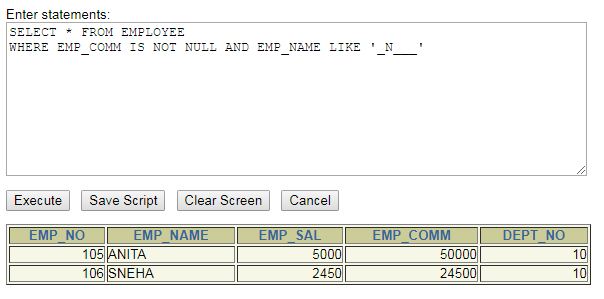
(5) Display the job name whose first three character in job id field is ‘FI\_’.



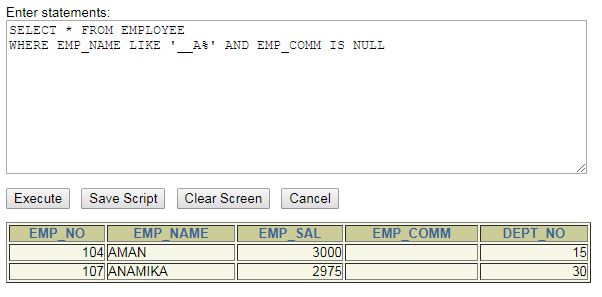
(6) Display the title/name of job whose last three character are ‘\_MGR’ and there maximum salary is greater than Rs 12000.



(7) Display the non-null values of employees and also employee name second character should be ‘n’ and string should be 5 character long.



(8) Display the null values of employee and also employee name’s third character should be ‘a’.

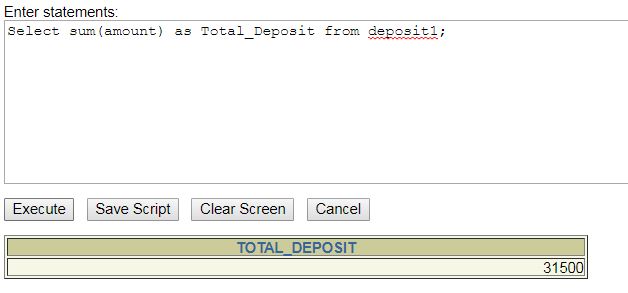


(9) What will be output if you are giving LIKE predicate as ‘%\\_%’ ESCAPE ‘\’

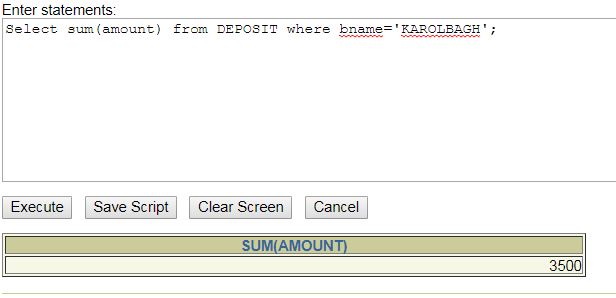


## To Perform various data manipulation commands, aggregate functions and sorting concept on all created tables.

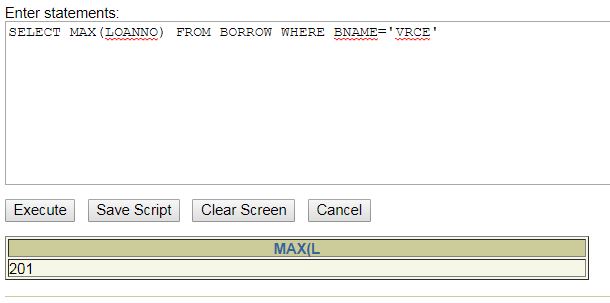
1. List total deposit from deposit.



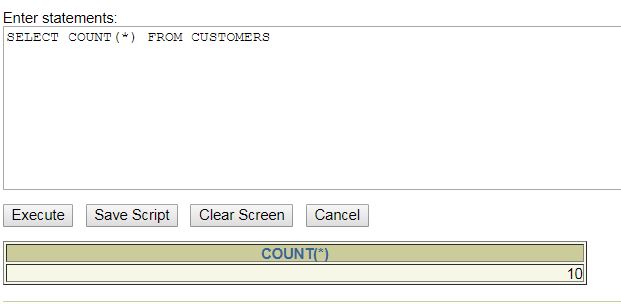
(2) List total loan from karolbagh branch



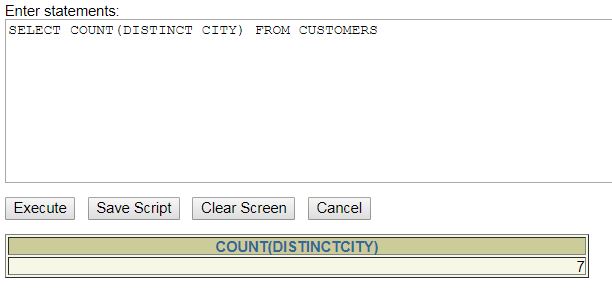
(3) Give maximum loan from branch vrce.



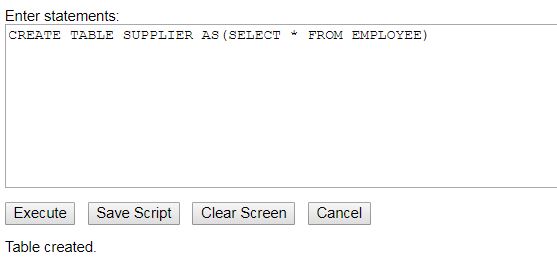
(4) Count total number of customers



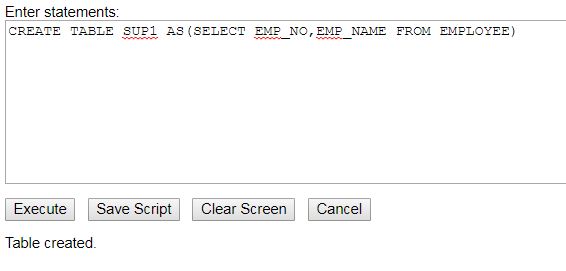
(5) Count total number of customer’s cities.



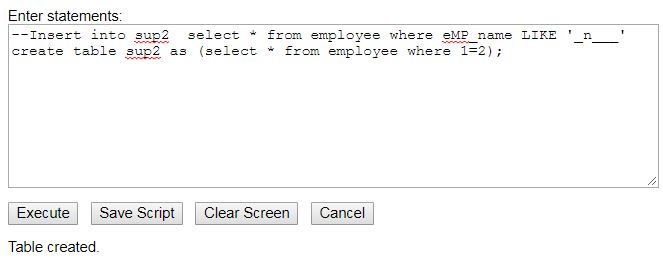
(6) Create table supplier from employee with all the columns.



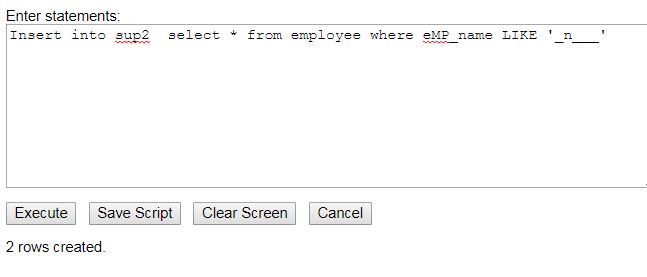
(7) Create table sup1 from employee with first two columns.



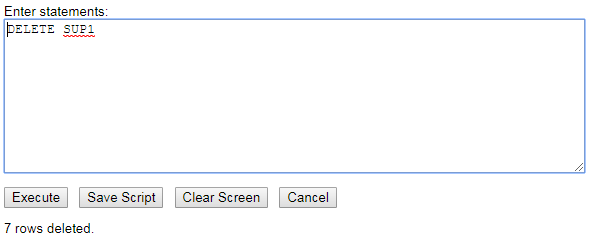
(8) Create table sup2 from employee with no data



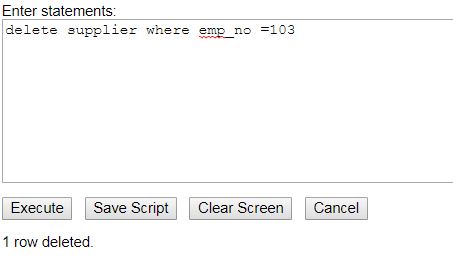
(9) Insert the data into sup2 from employee whose second character should be ‘n’ and string should be 5 characters long in employee name field.



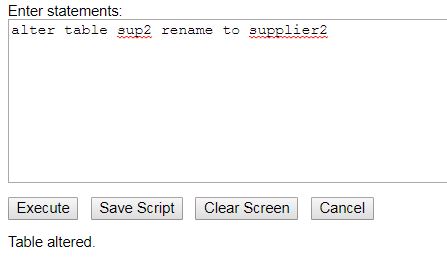
(10) Delete all the rows from sup1.



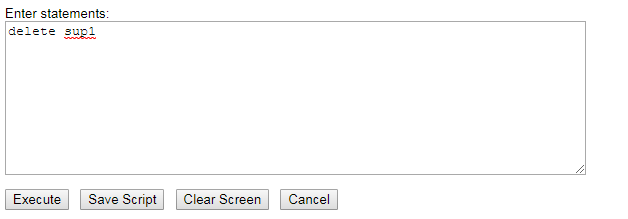
(11) Delete the detail of supplier whose sup\_no is 103.



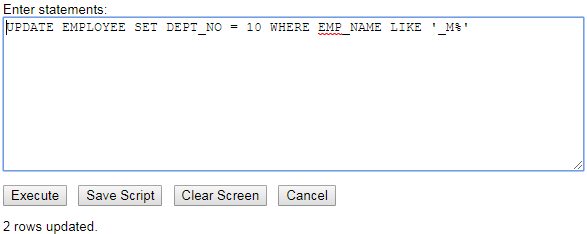
(12) Rename the table sup2.



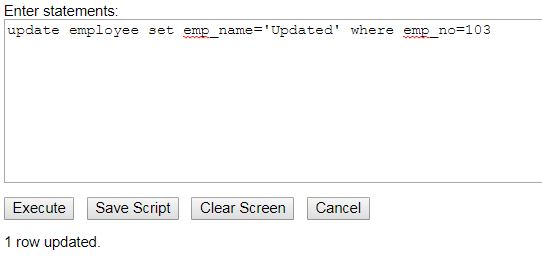
(13) Destroy table sup1 with all the data.



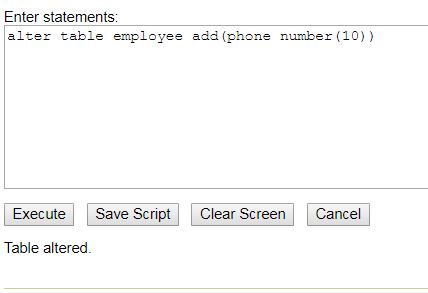
(14) Update the value dept\_no to 10 where second character of emp. name is ‘m’.



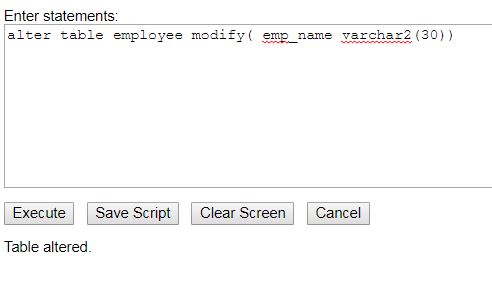
(15) Update the value of employee name whose employee number is 103.



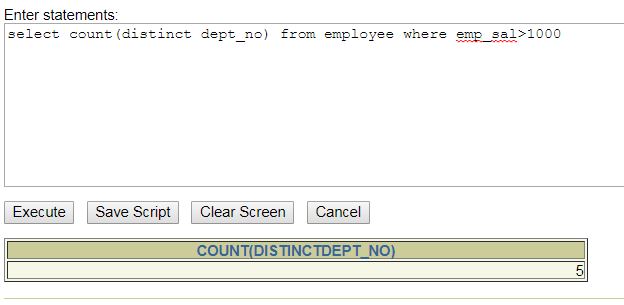
(16) Add one column phone to employee with size of column is 10.



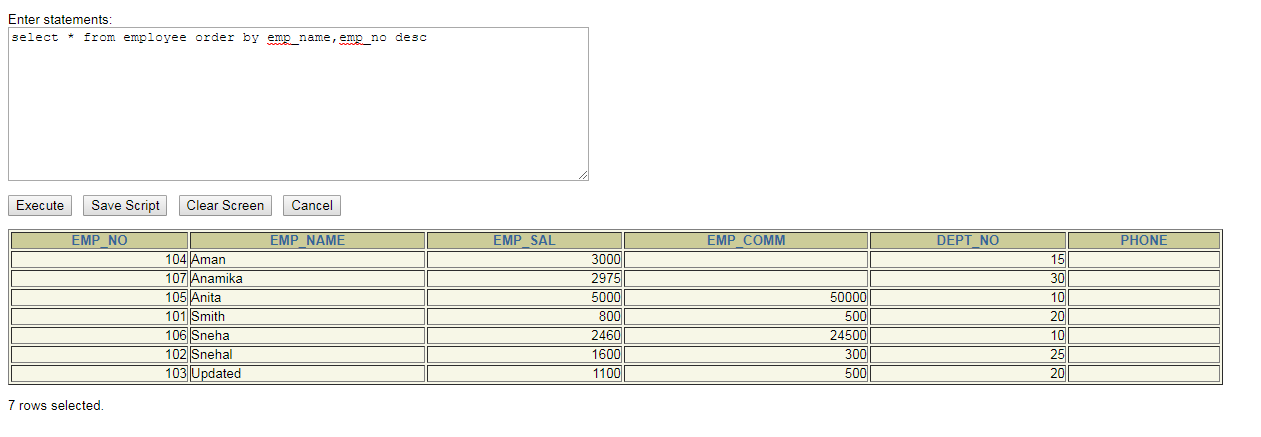
(17) Modify the column emp\_name to hold maximum of 30 characters.



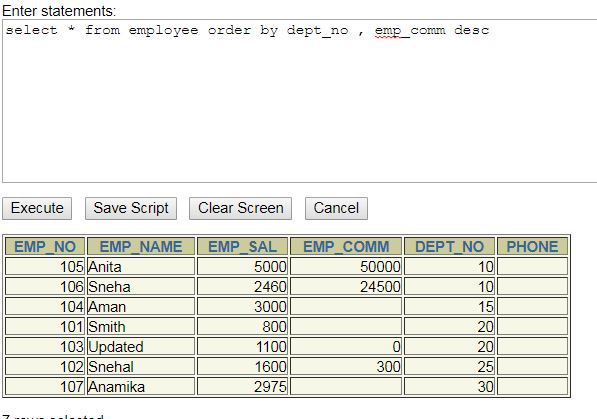
(18) Count the total no as well as distinct rows in dept\_no column with a condition of salary greater than 1000 of employee



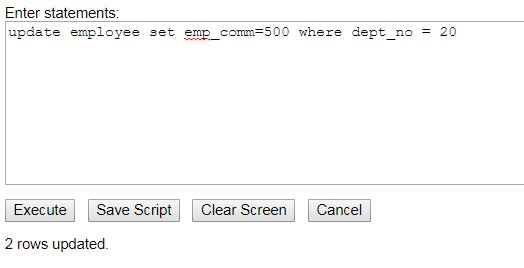
(19) Display the detail of all employees in ascending order, descending order of their name and no.



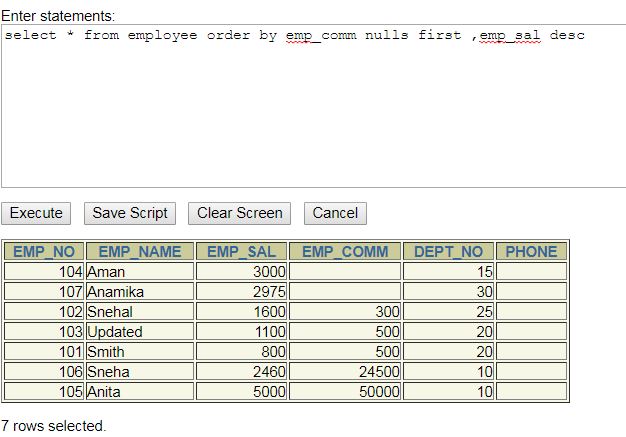
(20) Display the dept\_no in ascending order and accordingly display emp\_comm in descending order.



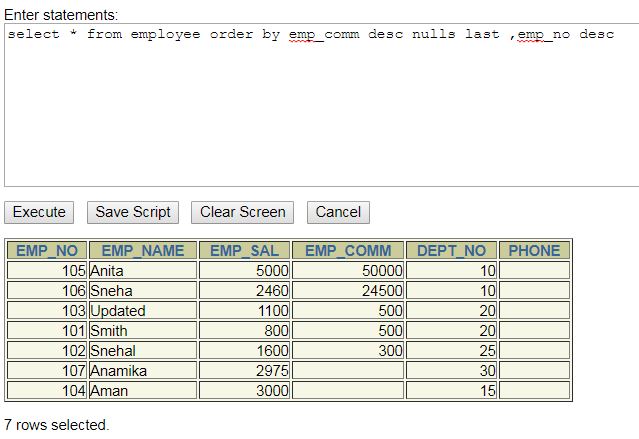
(21) Update the value of emp\_comm to 500 where dept\_no is 20.



(22) Display the emp\_comm in ascending order with null value first and accordingly sort employee salary in descending order.

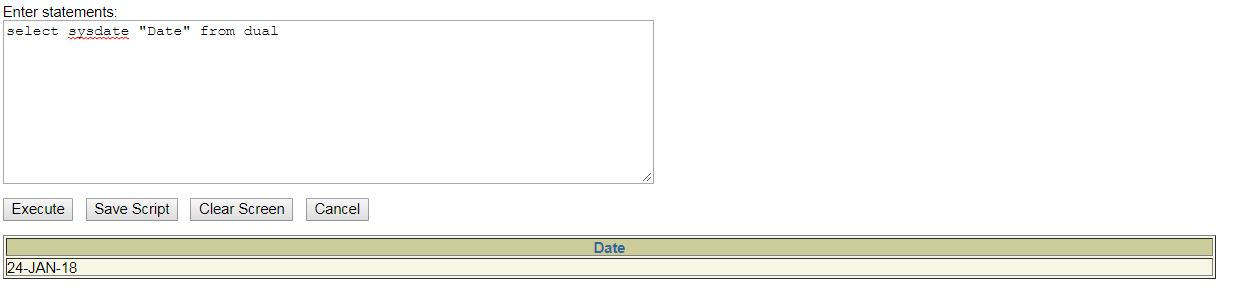


(23) Display the emp\_comm in ascending order with null value last and accordingly sort emp\_no in descending order.

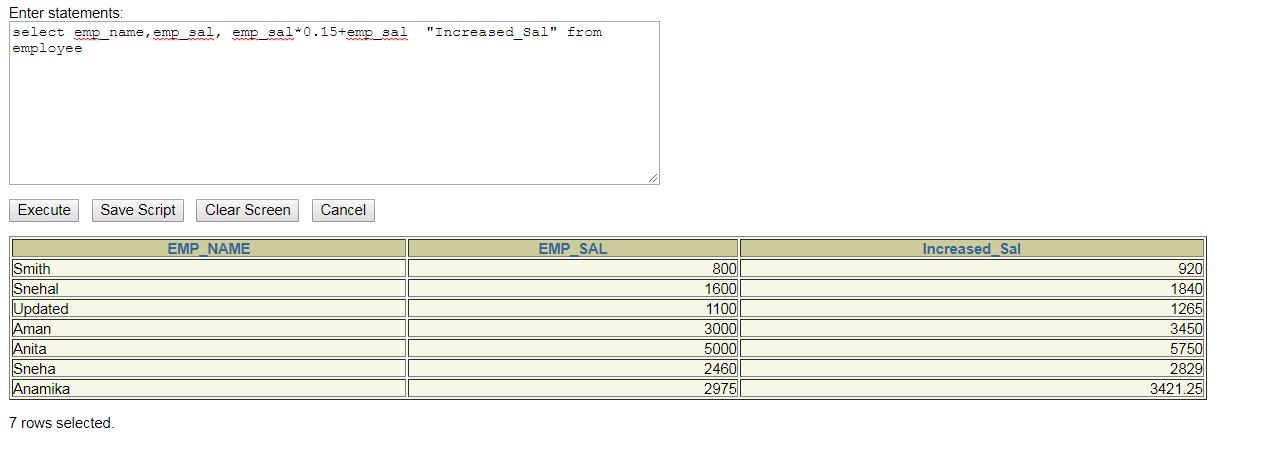


## To study Single-row functions.

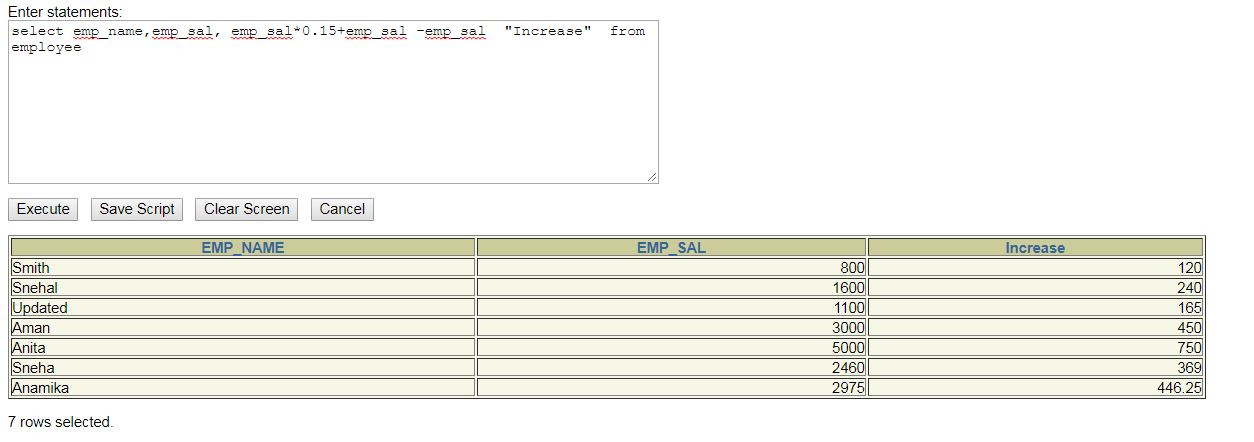
(1) Write a query to display the current date. Label the column Date



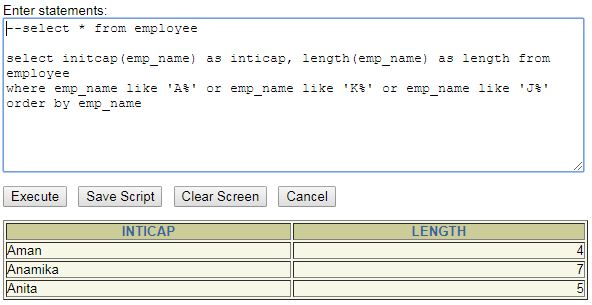
(2) For each employee, display the employee number, job, salary, and salary increased by 15% and expressed as a whole number. Label the column New Salary



(3) Modify your query no 4.(2) to add a column that subtracts the old salary from the new salary. Label the column Increase

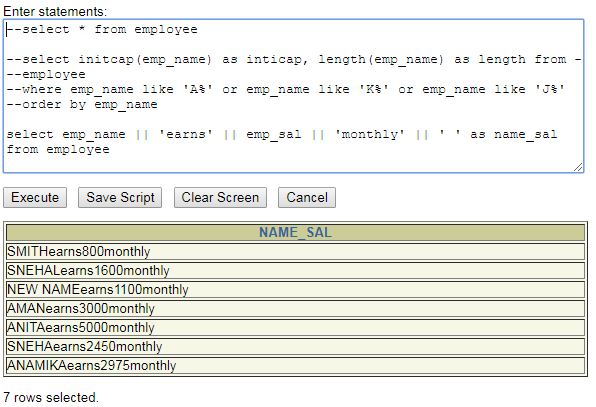


(4) Write a query that displays the employee’s names with the first letter capitalized and all other letters lowercase, and the length of the names, for all employees whose name starts with J, A, or M. Give each column an appropriate label. Sort the results by the employees’ last names.

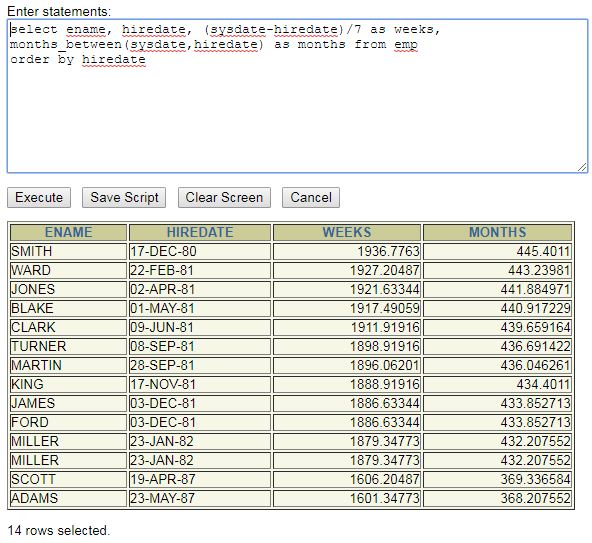


(5) Write a query that produces the following for each employee:

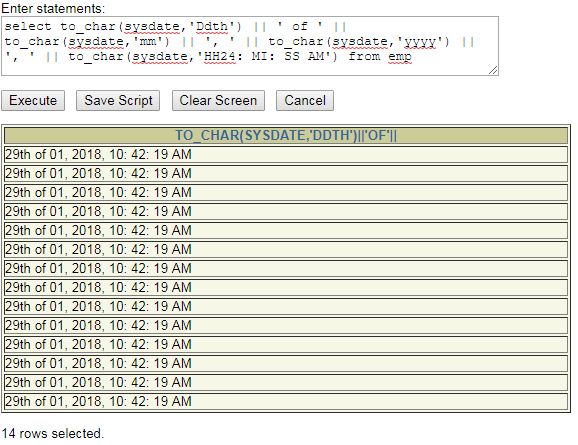
<employee last name> earns <salary> monthly



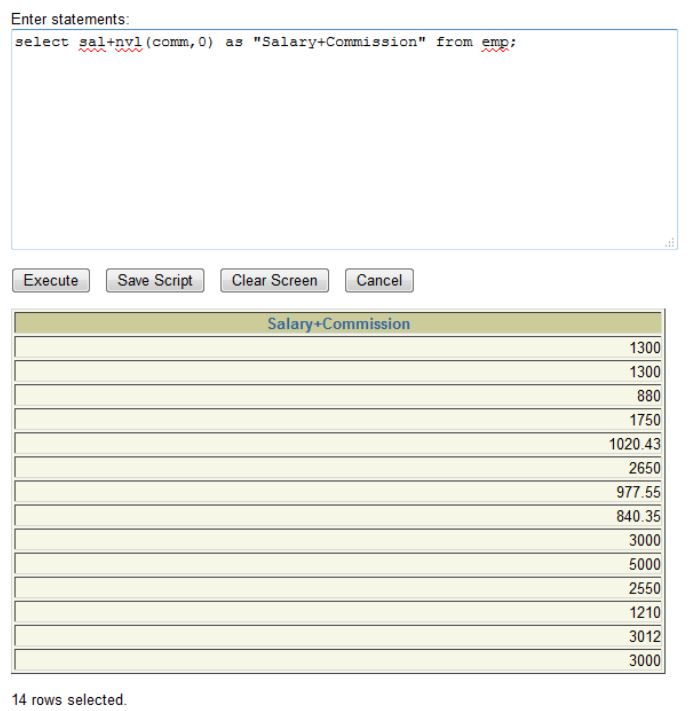
(6) Display the name, hire date, number of months employed and day of the week on which the employee has started. Order the results by the day of the week starting with Monday.



(7) Display the hiredate of emp in a format that appears as Seventh of June 1994 12:00:00 AM.



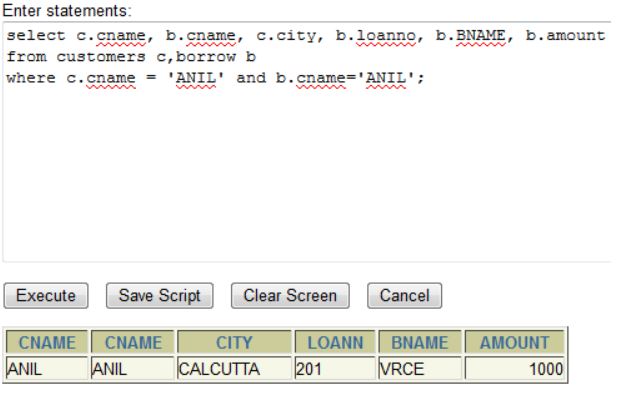
(8) Write a query to calculate the annual compensation of all employees (sal+comm.).



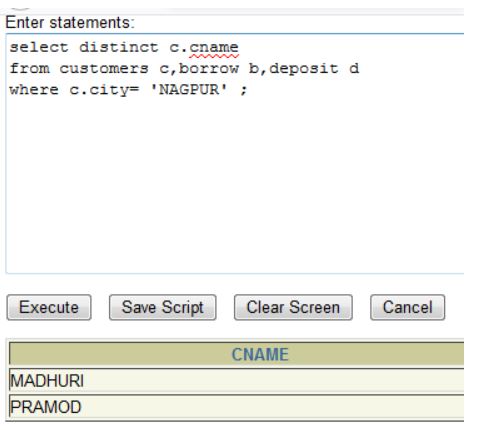
Practical 5

## Aim: Displaying data from Multiple Tables (join)

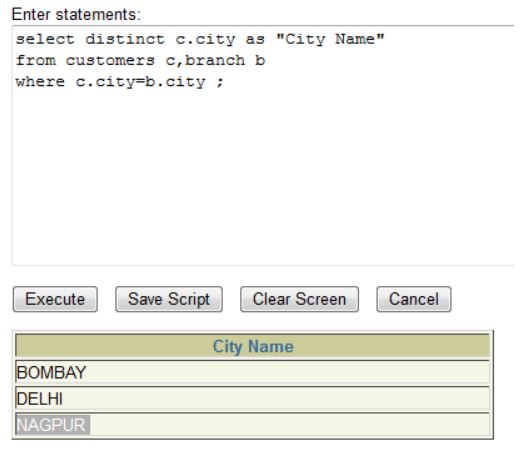
(1) Give details of customers ANIL.



(2) Give name of customer who are borrowers and depositors and having living city nagpur



(3) Give city as their city name of customers having same living branch.

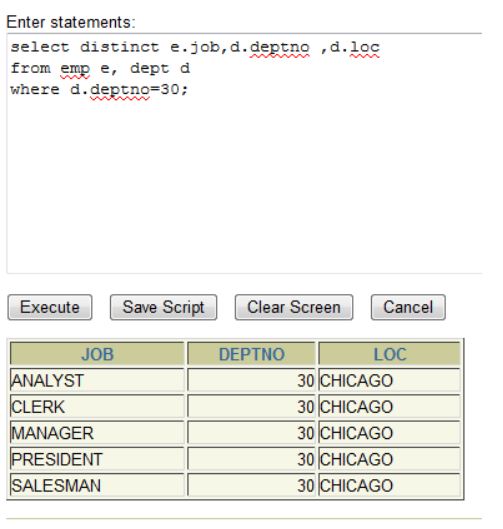


(4) Write a query to display the last name, department number, and department name for

all employees.



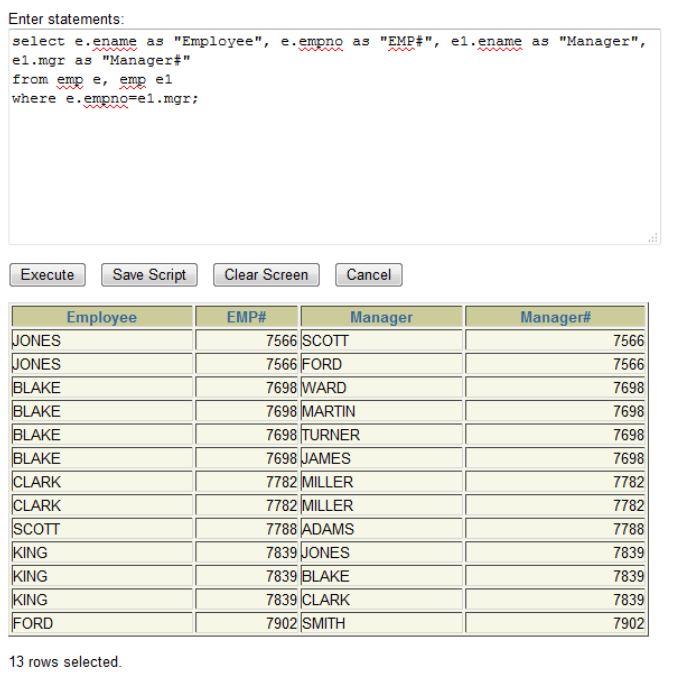
(5) Create a unique listing of all jobs that are in department 30. Include the location of the department in the output



(6) Write a query to display the employee name, department number, and department name for all employees who work in NEW YORK.



(7) Display the employee last name and employee number along with their manager’s last name and manager number. Label the columns Employee, Emp#, Manager, and Mgr#, respectively.



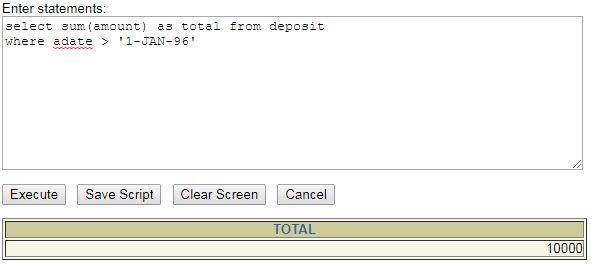
(8) Create a query to display the name and hire date of any employee hired after employee SCOTT.



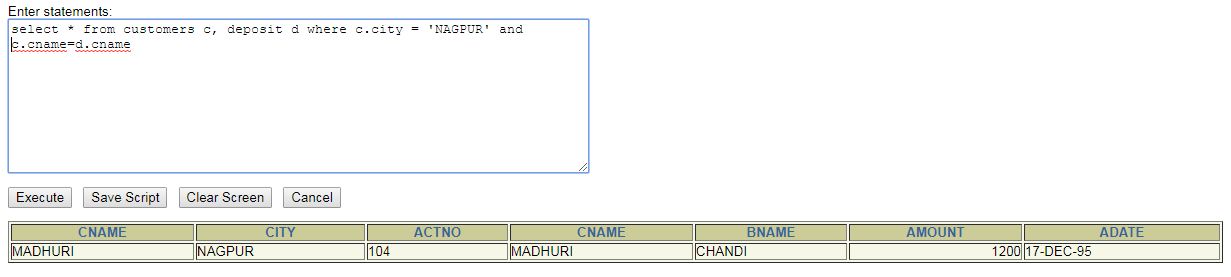
Practical 6

## Aim : To apply the concept of Aggregating Data using Group functions.

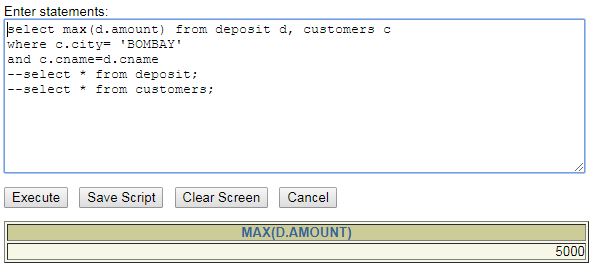
1. List total deposit of customer having account date after 1-jan-96.



(2) List total deposit of customers living in city Nagpur.

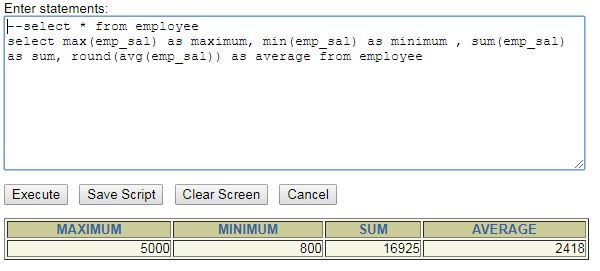


(3) List maximum deposit of customers living in bombay.

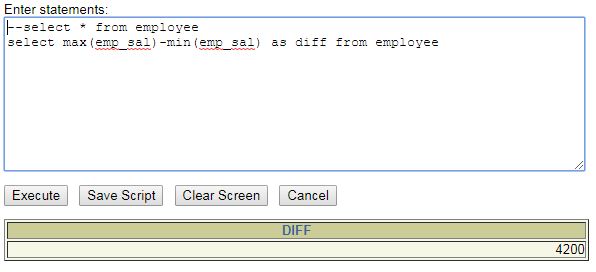


(4) Display the highest, lowest, sum, and average salary of all employees. Label the columns

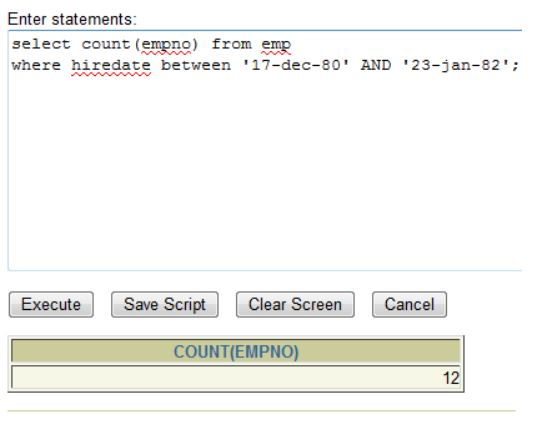
Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number.



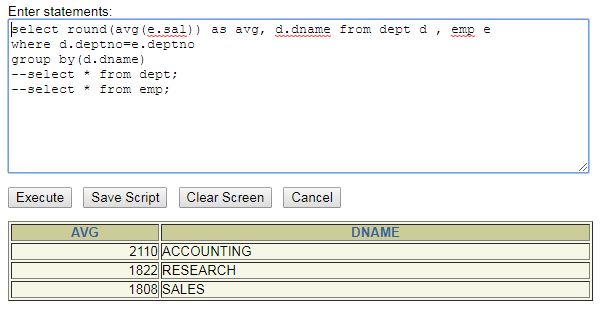
(5) Write a query that displays the difference between the highest and lowest salaries. Label the column DIFFERENCE.



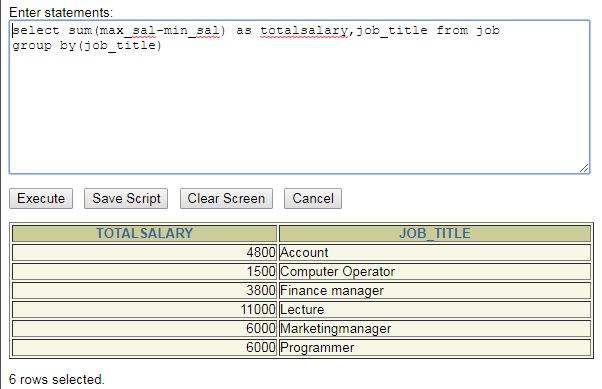
(6) Create a query that will display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998



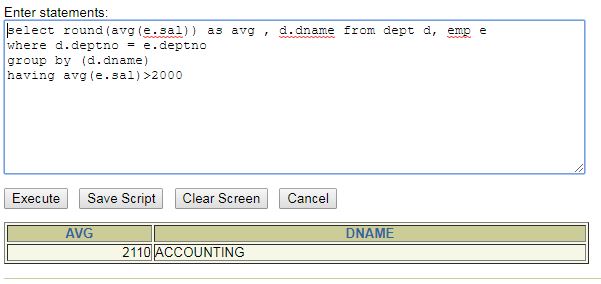
(7) Find the average salaries for each department without displaying the respective department numbers.



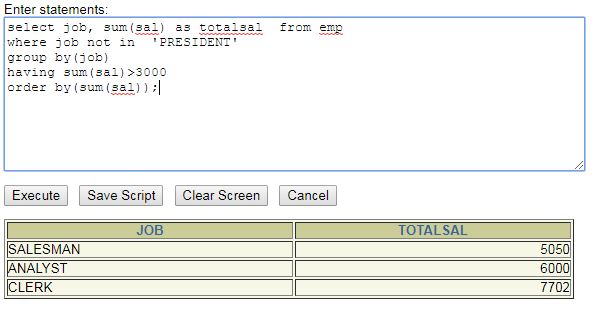
(8) Write a query to display the total salary being paid to each job title, within each department.



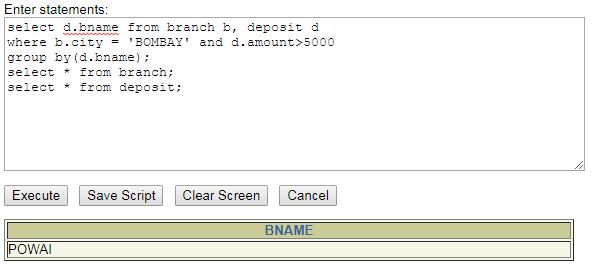
(9) Find the average salaries > 2000 for each department without displaying the respective department numbers.



(10) Display the job and total salary for each job with a total salary amount exceeding 3000, in which excludes president and sorts the list by the total salary



(11) List the branches having sum of deposit more than 5000 and located in city bombay.

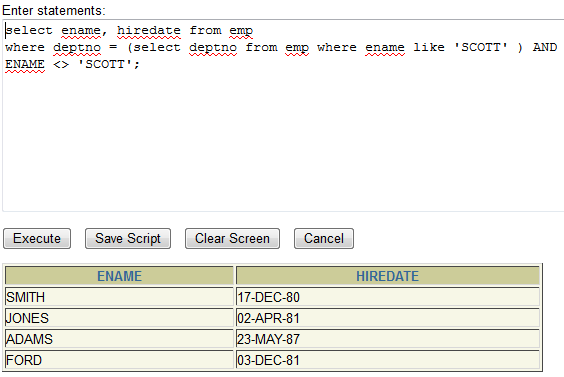


Practical 7

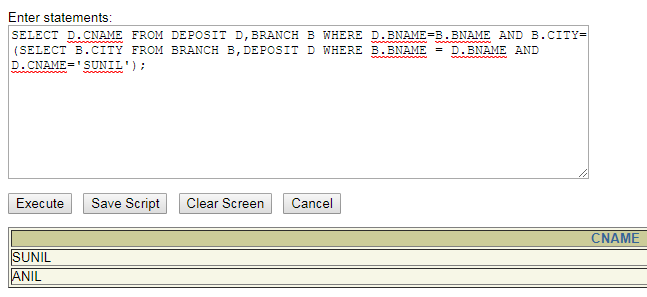
## Aim: To solve queries using the concept of sub query.

(1) Write a query to display the last name and hire date of any employee in the same

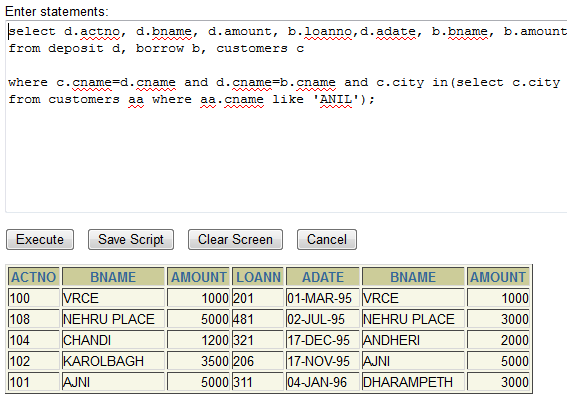
department as SCOTT. Exclude SCOTT.



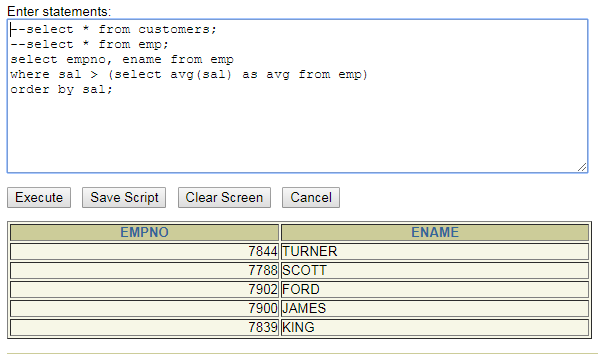
1. Give name of customers who are depositors having same branch city of mr. sunil.



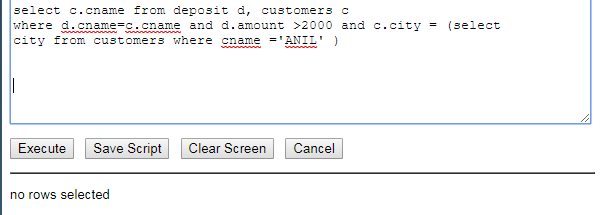
(3) Give deposit details and loan details of customer in same city where pramod is living.



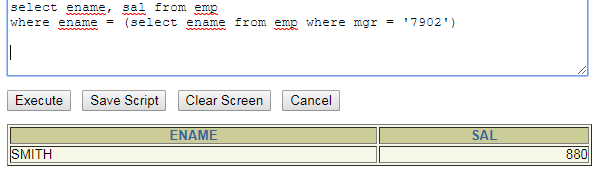
(4) Create a query to display the employee numbers and last names of all employees who earn more than the average salary. Sort the results in ascending order of salary.



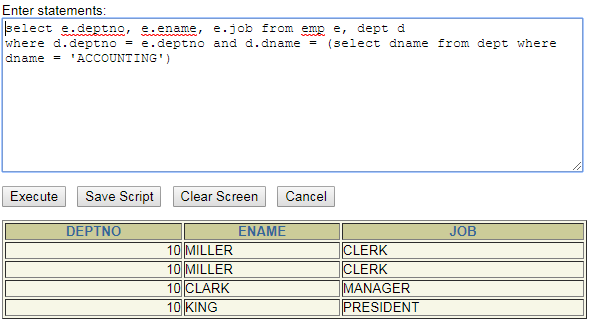
(5) Give names of depositors having same living city as mr. anil and having deposit amount greater than 2000



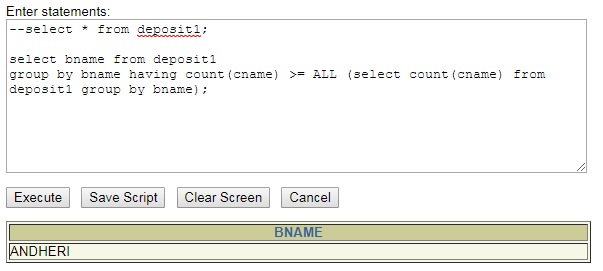
(6) Display the last name and salary of every employee who reports to ford.



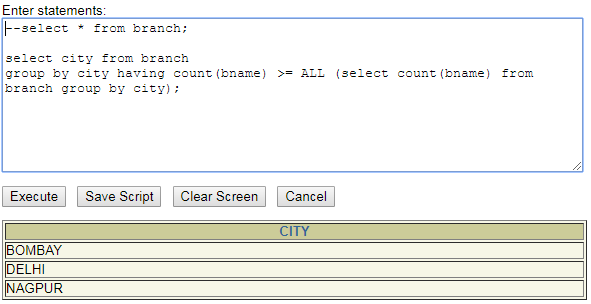
(7) Display the department number, name, and job for every employee in the Accounting department.



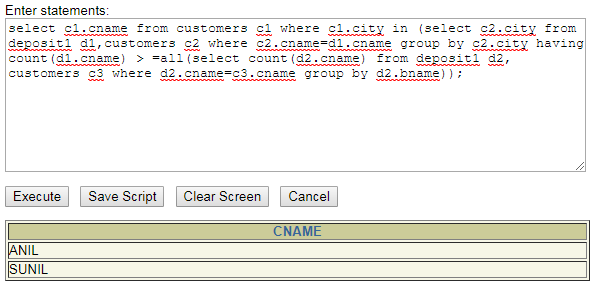
(8) List the name of branch having highest number of depositors.



(9) Give the name of cities where in which the maximum numbers of branches are located.



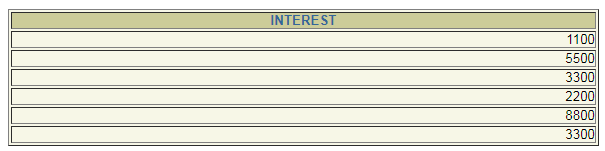
(10) Give name of customers living in same city where maximum depositors are located.



Practical 8

## Aim : Manipulating Data

(1) Give 10% interest to all depositors.

select amount\*0.1+amount Interest from borrow

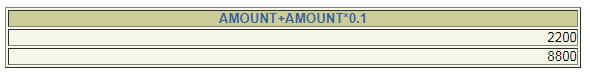
(2) Give 10% interest to all depositors having branch vrce

select amount\*0.1+amount Interest from borrow where bname='VRCE'



(3) Give 10% interest to all depositors living in nagpur and having branch city bombay.

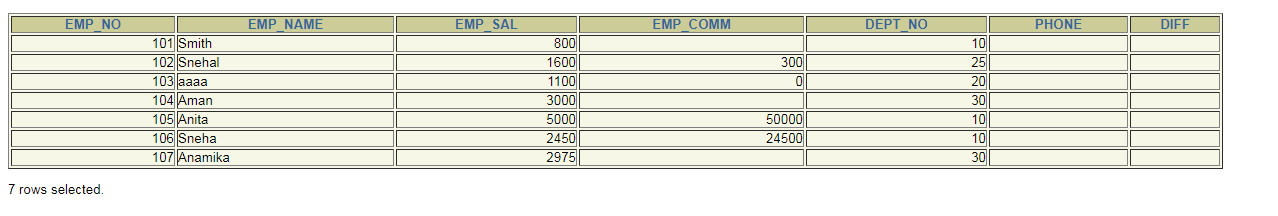
select amount+amount\*0.1 from borrow,branch,customers where branch.bname=borrow.bname and borrow.cname=customers.cname and customers.city='NAGPUR' and branch.city='BOMBAY';



(4) Write a query which changes the department number of all employees with empno 7788’s job to employee 7844’current department number.

update employee set dept\_no = ( select dept\_no from employee where emp\_no=107) where emp\_no=104;

select \* from employee

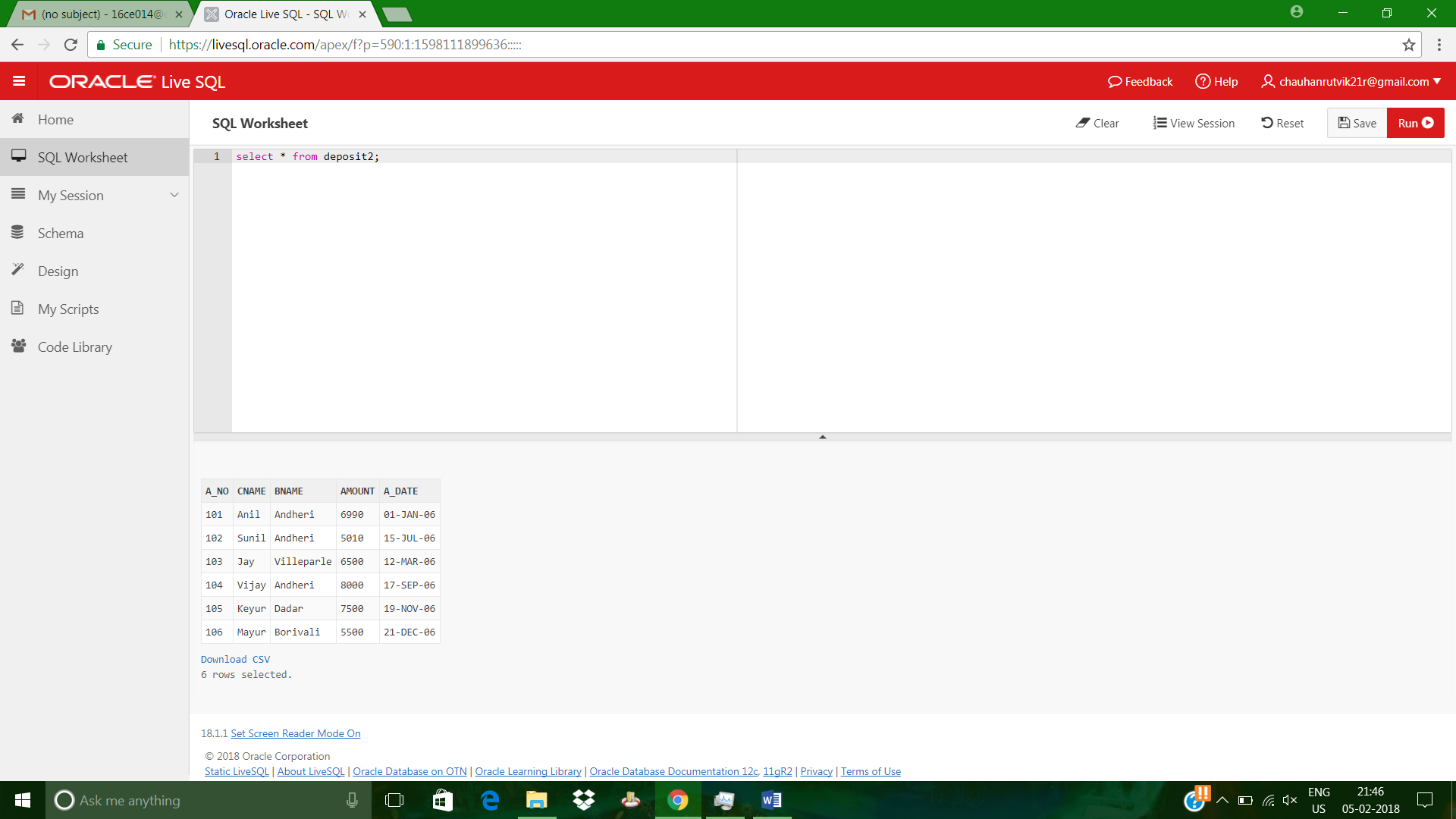


(5) Transfer 10 Rs from account of anil to sunil if both are having same branch.

update deposit2 set amount = amount-10 where cname='Anil' and bname in(select d1.bname from deposit2 d1 where d1.cname='Sunil');

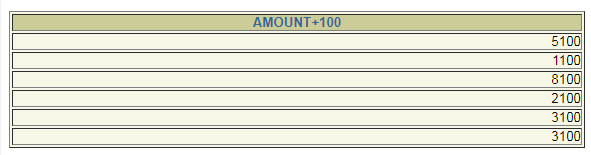
update deposit2 set amount = amount+10 where cname='Sunil' and bname in(select d2.bname from deposit2 d2 where d2.cname='Anil');

select \* from deposit2;



(6) Give 100 Rs more to all depositors if they are maximum depositors in their respective branch.

select amount+100 from borrow group by bname,amount having amount=max(amount);



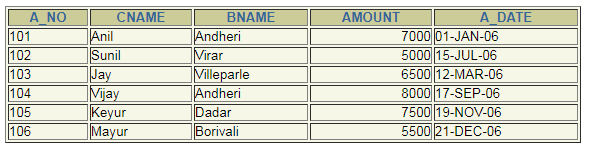
(7) Delete depositors of branches having number of customers between 1 to 3.

delete from deposit2 where a\_no in (101,102,103);



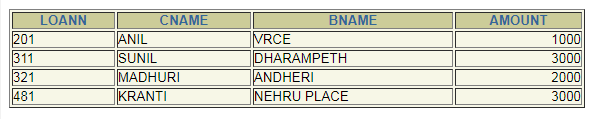
(8) Delete deposit of vijay.

delete from deposit2 where cname='Vijay';



(9) Delete borrower of branches having average loan less than 1000.

delete from borrow where amount<(select avg(amount) from borrow where amount>4000);



Practical 9

## AIM : To perform basic PL/SQL blocks

**Theory:**

SQL is a very powerful, set-oriented language whose sole purpose is to manipulate the contents of relational databases. If you write applications built on Oracle Database, you (or someone writing code at a lower level in the technology stack) must be executing SQL statements to retrieve data from or change data in that database. Yet SQL cannot be used to implement all business logic and end-user functionality needed in our applications. That brings us to PL/SQL. PL/SQL stands for Procedural Language/Structured Query Language. PL/SQL offers a set of procedural commands (IF statements, loops, assignments), organized within blocks (explained below), that complement and extend the reach of SQL. It is certainly possible to build applications on top of SQL and Oracle Database without using PL/SQL. Utilizing PL/SQL to perform database-specific operations, most notably SQL statement execution, offers several advantages, though, including tight integration with SQL, improved performance through reduced network traffic, and portability (PL/SQL programs can run on any Oracle Database instance). Thus, the front-end code of many applications executes both SQL statements and PL/SQL blocks, to maximize performance while improving the maintainability of those applications. PL/SQL is a block-structured language. A PL/SQL block is defined by the keywords DECLARE, BEGIN, EXCEPTION, and END, which break up the block into three sections: Declarative: statements that declare variables, constants, and other code elements, which can then be used within that block Executable: statements that are run when the block is executed Exception handling: a specially structured section you can use to “catch,” or trap, any exceptions that are raised when the executable section runs Only the executable section is required. You don’t have to declare anything in a block, and you don’t have to trap exceptions raised in that block. A block itself is an executable statement, so you can nest blocks within other blocks.

**Q.** Write a PL-SQL block for checking weather a given year is a Leap year or not

**set serveroutput on;**

**declare**

**a number(5);**

**begin**

**a:=&a;**

**dbms\_output.put\_line('hello');**

**dbms\_output.put\_line('a='||&a);**

**if ((mod(a,4)=0 and mod(a,100)!=0) or mod(a,400)=0) then**

**dbms\_output.put\_line('leap year');**

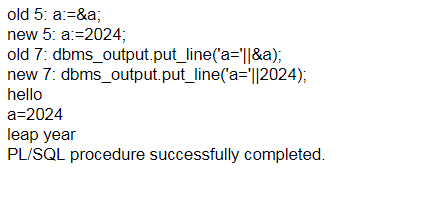
**else**

**dbms\_output.put\_line('not leap year');**

**end if;**

**end;**

**Output:**



Practical 10

**AIM : To perform the concept of loop**

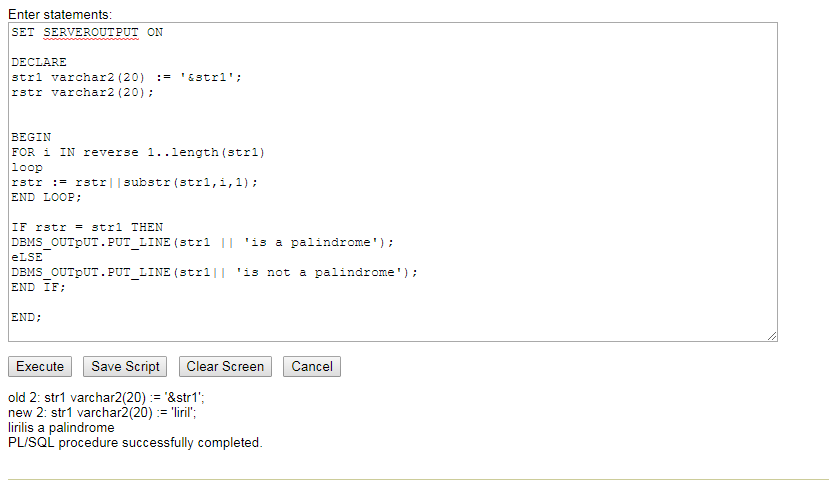
**Q.** Find out whether given string is palindrome or not using For, While and Simple

Loop

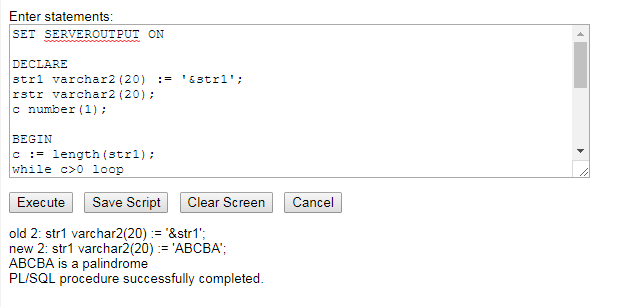
**Using simple loop**



**For loop**



**While loop**



Practical 11

**Aim: To understand the concept of “select into” and “% type” attribute.**

**Theory**:

The SELECT INTO statement copies data from one table and inserts it into a new table.

The number of columns and data type of column must be same.

Syntax of a SELECT INTO STATEMENT

SELECT column\_name(s)  
INTO new\_table  
FROM table\_name;

Note: WHERE condition is optional to use along with SELECT INTO Clause.

SQL SELECT INTO STATEMENT Example:

Database table “Employee”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Employee ID | Employee Name | Age | Gender | Location | Salary |
| 1001 | Henry | 54 | Male | New York | 100000 |
| 1002 | Tina | 36 | Female | Moscow | 80000 |
| 1003 | John | 24 | Male | London | 40000 |
| 1004 | Mile | 31 | Male | London | 70000 |
| 1005 | Tara | 26 | Female | Moscow | 50000 |
| 1006 | Sohpie | 29 | Female | London | 60000 |

Below is an example of a sql statement to create a copy of “Employee” table, use of \* selects all rows.

If you want to copy certain columns then declare the column name(s)  
*The user can alternatively specify number of rows or % of rows*

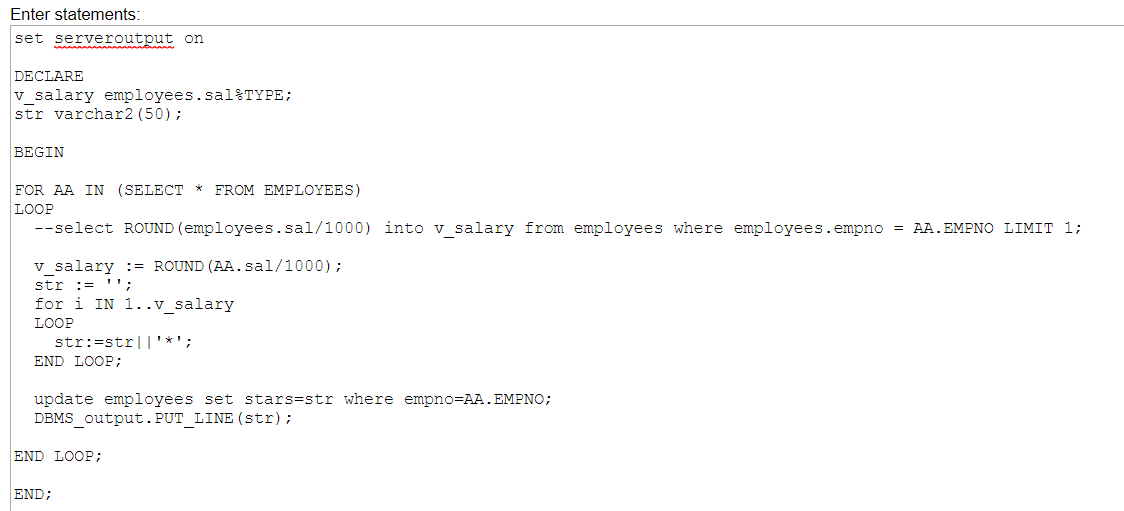
SELECT \*   
INTO Copy\_Employee  
FROM Employee;

Command will create a table similar to “Employee” table. This SELECT INTO statement is often used for creating a backup copy of a table**.**

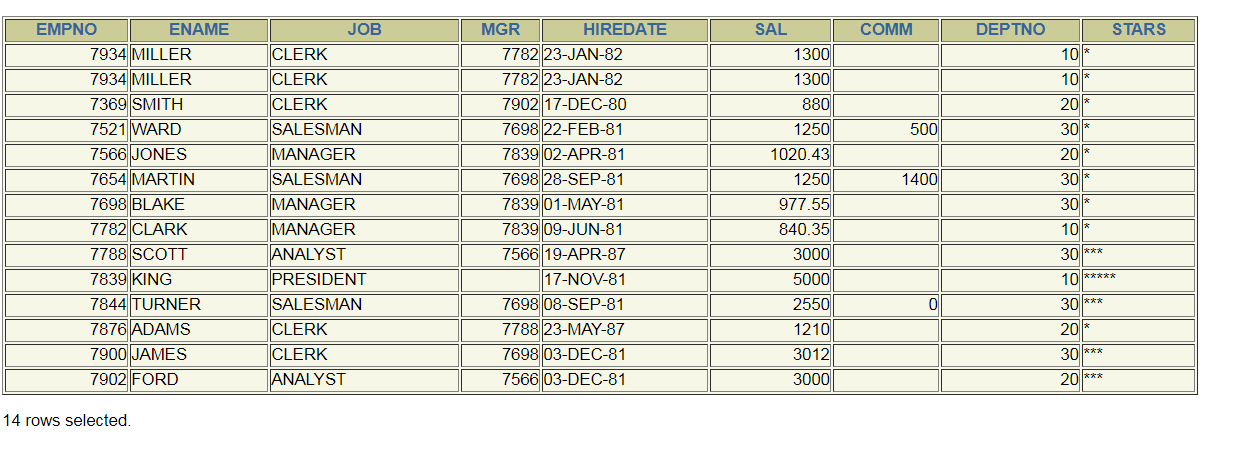
**Q**. Create an EMPLOYEES table that is a replica of the EMP table. Add a new column, STARS, of VARCHAR2 data type and length of 50 to the EMPLOYEES table for storing asterisk (\*).

Create a PL/SQL block that rewards an employee by appending an asterisk in the STARS column for every Rs1000/- of the employee’s salary. For example, if the employee has a salary amount of Rs8000/-, the string of asterisks should contain eight asterisks. If the employee has a salary amount of Rs12500/-, the string of asterisks should contain 13 asterisks.

Update the STARS column for the employee with the string of asterisks.



**OUTPUT:**



Practical 12

## Aim: To perform the concept of cursor

**Theory:**

A cursor is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the active set.

You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors −

* Implicit cursors
* Explicit cursors

*Implicit Cursors*

Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement. Programmers cannot control the implicit cursors and the information in it.

Whenever a DML statement (INSERT, UPDATE and DELETE) is issued, an implicit cursor is associated with this statement. For INSERT operations, the cursor holds the data that needs to be inserted. For UPDATE and DELETE operations, the cursor identifies the rows that would be affected.

In PL/SQL, you can refer to the most recent implicit cursor as the **SQL cursor**, which always has attributes such as **%FOUND, %ISOPEN, %NOTFOUND**, and **%ROWCOUNT**.

*Explicit Cursors*

Explicit cursors are programmer-defined cursors for gaining more control over the **context area**. An explicit cursor should be defined in the declaration section of the PL/SQL Block. It is created on a SELECT Statement which returns more than one row.

The syntax for creating an explicit cursor is −

CURSOR cursor\_name IS select\_statement;

Working with an explicit cursor includes the following steps −

* Declaring the cursor for initializing the memory
* Opening the cursor for allocating the memory
* Fetching the cursor for retrieving the data
* Closing the cursor to release the allocated memory

(a) Display all the information of EMP table using %ROWTYPE.



(b) Create a PL/SQL block that does the following:

In a PL/SQL block, retrieve the name, salary, and MANAGER ID of the employees working in the particular department. Take Department Id from user.

If the salary of the employee is less than 1000 and if the manager ID is either 7902 or 7839, display the message <<last\_name>> Due for a raise. Otherwise, display the message <<last\_name>> Not due for a raise.

