CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

FACULTY OF TECHNOLOGY AND ENGINEERING

Devang Patel Institute of Advance Technology & Research

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CE246 Database Management System

Semester: IV

Academic Year: 2019-20

PRACTICAL LIST

Sr. No.	Aim of the Practical	Date	Page NO.	Remark
1	**Introduction to Oracle Architecture.**	4/12/19	12	
2	**To study DDL-create and DML-insert commands**. **(i)** Create tables according to the following definition. ● CREATE TABLE DEPOSIT (ACTNO VARCHAR2(5), CNAME VARCHAR2(18), BNAME VARCHAR2(18), AMOUNT NUMBER (8,2), ADATE DATE); ● CREATE TABLE BRANCH (BNAME VARCHAR2(18), CITY VARCHAR2(18)); ● CREATE TABLE CUSTOMERS (CNAME VARCHAR2(19), CITY VARCHAR2(18)); ● CREATE TABLE BORROW (LOANNO VARCHAR2(5), CNAME VARCHAR2(18), BNAME VARCHAR2(18), AMOUNT NUMBER (8,2)); **(ii)** Insert the data as shown below. **DEPOSIT**	9/12/19	14	

ACTNO	**CNAME**	**BNAME**	**AMOUNT**	**ADATE**
100	ANIL	VRCE	1000.00	1-MAR-95
101	SUNIL	AJNI	5000.00	4-JAN-96
102	MEHUL	KAROLBAGH	3500.00	17-NOV-95

104	MADHURI	CHANDI	1200.00	17-DEC-95
105	PRMOD	M.G.ROAD	3000.00	27-MAR-96
106	SANDIP	ANDHERI	2000.00	31-MAR-96
107	SHIVANI	VIRAR	1000.00	5-SEP-95
108	KRANTI	NEHRU PLACE	5000.00	2-JUL-95
109	MINU	POWAI	7000.00	10-AUG-95

BRANCH

VRCE	NAGPUR
AJNI	NAGPUR
KAROLBAGH	DELHI
CHANDI	DELHI
DHARAMPETH	NAGPUR
M.G.ROAD	BANGLORE
ANDHERI	BOMBAY
VIRAR	BOMBAY
NEHRU PLACE	DELHI
POWAI	BOMBAY

CUSTOMERS

ANIL	CALCUTTA
SUNIL	DELHI
MEHUL	BARODA
MANDAR	PATNA
MADHURI	NAGPUR
PRAMOD	NAGPUR
SANDIP	SURAT
SHIVANI	BOMBAY
KRANTI	BOMBAY
NAREN	BOMBAY

BORROW

LOANNO	CNAME	BNAME	AMOUNT
201	ANIL	VRCE	1000.00
206	MEHUL	AJNI	5000.00
311	SUNIL	DHARAMPETH	3000.00
321	MADHURI	ANDHERI	2000.00
375	PRMOD	VIRAR	8000.00
481	KRANTI	NEHRU PLACE	3000.00

From the above given tables perform the following queries:
(1) Describe deposit, branch.
(2) Describe borrow, customers.
(3) List all data from table DEPOSIT.
(4) List all data from table BORROW.
(5) List all data from table CUSTOMERS.
(6) List all data from table BRANCH.
(7) Give account no and amount of depositors.
(8) Give name of depositors having amount greater than 4000.
(9) Give name of customers who opened account after date '1-12-96'.
(10) Give name of city where branch karolbagh is located.
(11) Give account no and amount of customer having account opened between date 1-12-96 and 1-6-96.
(12) Give names of depositors having account at VRCE.

3	**Create the below given table and insert the data accordingly**.	11/12 /19	24	

Create Table Job (job_id, job_title, min_sal, max_sal)

COLUMN NAME	DATA TYPE
job_id	Varchar2(15)
job_title	Varchar2(30)
min_sal	Number(7,2)
max_sal	Number(7,2)

Create table **Employee** (emp_no, emp_name, emp_sal, emp_comm, dept_no)

COLUMN NAME	DATA TYPE
emp_no	Number(3)
emp_name	Varchar2(30)
emp_sal	Number(8,2)
emp_comm	Number(6,1)
dept_no	Number(3)

Create table **deposit**(a_no,cname,bname,amount,a_date).

COLUMN NAME	DATA TYPE
a_no	Varchar2(5)
cname	Varchar2(15)
bname	Varchar2(10)
amount	Number(7,2)
a_date	Date

Create table **borrow** (loanno, cname, bname, amount).

COLUMN NAME	DATA TYPE
loanno	Varchar2(5)
cname	Varchar2(15)
bname	Varchar2(10)
amount	Varchar2(7,2)

Insert following values in the table **Employee**.

emp_no	**emp_name**	**emp_sal**	**emp_comm**	**dept _no**
101	Smith	800		20
102	Snehal	1600	300	25
103	Adama	1100	0	20

104	Aman	3000		15
105	Anita	5000	50,000	10
106	Sneha	2450	24,500	10
107	Anamika	2975		30

Insert following values in the table **ob**.

job_id	**job_name**	**min_sal**	**max_sal**
IT_PROG	Programmer	4000	10000
MK_MGR	Marketing manager	9000	15000
FI_MGR	Finance manager	8200	12000
FI_ACC	Account	4200	9000
LEC	Lecturer	6000	17000
COMP_OP	Computer Operator	1500	3000

Insert following values in the table **deposit**.

A_no	**cname**	**Bname**	**Amount**	**date**
101	Anil	andheri	7000	01-jan-06
102	sunil	virar	5000	15-jul-06
103	jay	villeparle	6500	12-mar-06
104	vijay	andheri	8000	17-sep-06
105	keyur	dadar	7500	19-nov-06
106	mayur	borivali	5500	21-dec-06

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.
(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**. **To study various options of 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 who’s last three character are ‘**_MGR**’ and their 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 ‘\’			
4	**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.	16/12/19	32	

	(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.			
5	**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 (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.).	18/12/19	40	

6	**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.	23/12 /19	44	

7	**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.	30/12 /19	41	

8	**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 (2) 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.	06/01/20	51	
9	**Manipulating Data** (1) Give 10% interest to all depositors. (2) Give 10% interest to all depositors having branch vrce (3) Give 10% interest to all depositors living in nagpur and having 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. (5) Transfer 10 Rs from account of anil to sunil if both are having same branch. (6) Give 100 Rs more to all depositors if they are maximum depositors in their respective branch. (7) Delete depositors of branches having number of customers between 1 to 3. (8) Delete deposit of vijay. (9) Delete borrower of branches having average loan less than 1000.	08/01/20	55	

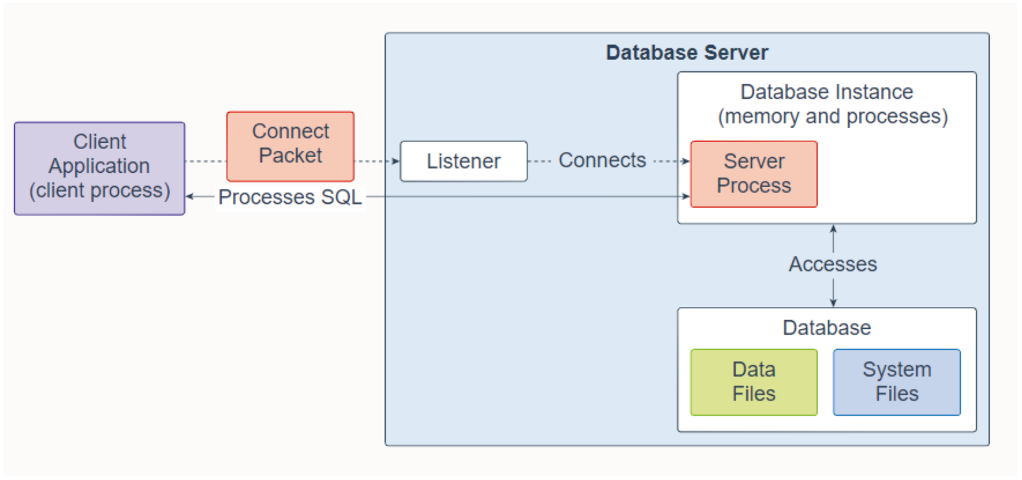
10	**To perform basic PL/SQL blocks** Write a PL-SQL block for checking weather a given year is a Leap year or not	22/01/20	59	
11	**To perform the concept of loop** Find out whether given string is palindrome or not using for, While and Simple Loop.	05/02/20	60	
12	**To understand the concept of "select into" and "% type" attribute**. 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.	17/02/20	62	
13	**To perform the concept of cursor** (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.	24/02/20	64	
14	**To perform the concept of trigger** Write a PL/SQL block to update the salary where deptno is 10. Generate trigger that will store the original record in other table before updation take place	26/02/20	67	

15	**To perform the concept of function and procedure** Write a PL/SQL block to update the salary of employee specified by empid. If record exist, then update the salary otherwise display appropriate message. Write a function as well as procedure for updating salary.	02/03 /20	69	
16	**To perform the concept of exception handler** Write a PL/SQL block that will accept the employee code, amount and operation. Based on specified operation amount is added or deducted from salary of said employee. Use user defined exception handler for handling the exception.	09/03 /20	72	
17	**To perform the concept of package** Create and invoke a package that contains private and public constructs.	11/03 /20	74	

Sr No.	PRACTICALS

1. **Aim:-Introduction to Oracle Architecture.**

Theory:



An Oracle Database consists of at least one database instance and one database. The database instance handles memory and processes. The database consists of physical files called data files, and can be a non-container database or a multitenant container database. An Oracle Database also uses several database System files during its operation.
A single-instance database architecture consists of one database instance and one database. A one-to-one relationship exists between the database and the database instance. Multiple single-instance databases can be installed on the same server machine. There are separate database instances for each database. This configuration is useful to run different versions of Oracle Database on the same machine

Questions and Answers / Case Study :-

1. Need of DBMS?

A. Database is a collection of inter-related data which helps in efficient retrieval, insertion and deletion of data from database and organizes the data in the form of tables, views, schemas, reports etc. For Example, university database organizes the data about students, faculty, and admin staff etc. which helps in efficient retrieval, insertion and deletion of data from it.

2. What is DBMS?

A. The software which is used to manage database is called Database Management System (DBMS). For Example, MySQL, Oracle etc. are popular commercial DBMS used in different applications.

3. What are different languages used in DBMS?

A.
i) DDL- Data Definition Language

	ii)DML- Data Manipulation Language iii)DQL- Data Query Language iv)DCL- Data Control Language V)TCL- Transaction Control Language

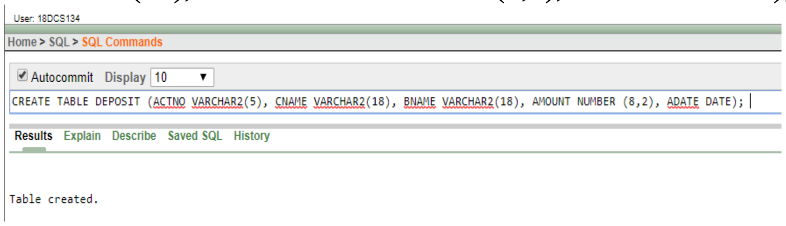
2. **Aim:-**

To study DDL-create and DML-insert commands.

i)Create tables according to the following definition.

<u>Query:</u>

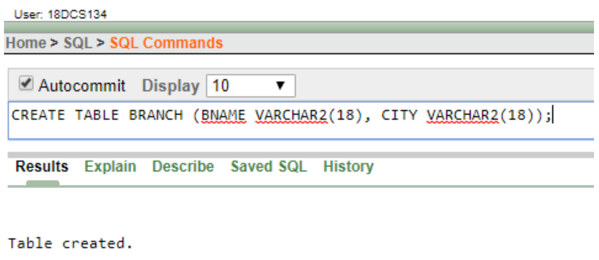
CREATE TABLE DEPOSIT (ACTNO VARCHAR2(5), CNAME VARCHAR2(18), BNAME VARCHAR2(18), AMOUNT NUMBER (8,2), ADATE DATE);


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

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

```
Table created.
```

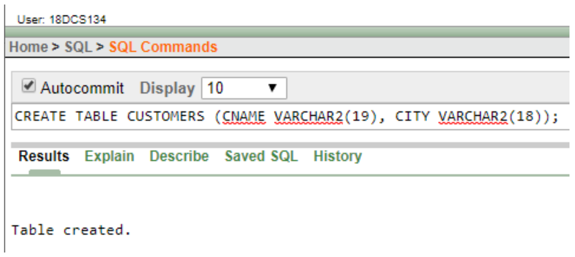
CREATE TABLE BRANCH (BNAME VARCHAR2(18), CITY VARCHAR2(18));


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

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

```
Table created.
```

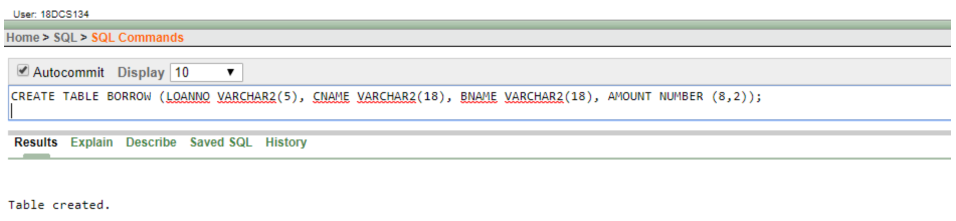
CREATE TABLE CUSTOMERS (CNAME VARCHAR2(19), CITY VARCHAR2(18));


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

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

```
Table created.
```

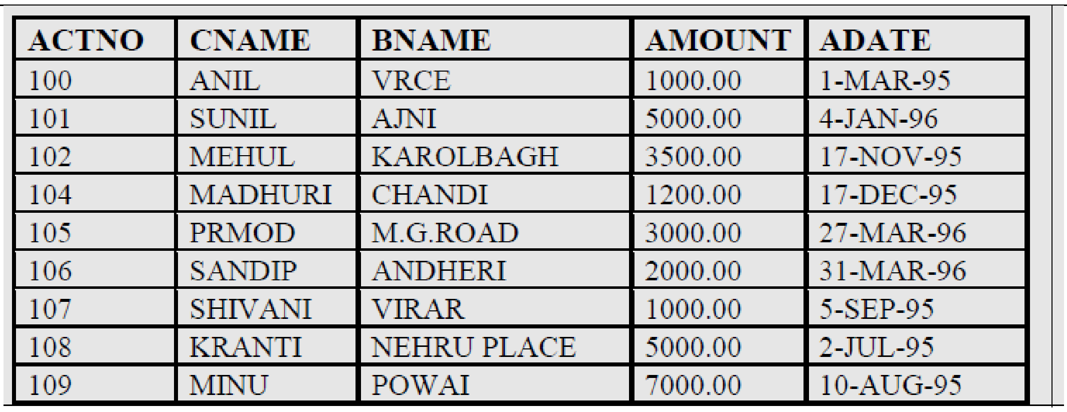
CREATE TABLE BORROW (LOANNO VARCHAR2(5), CNAME VARCHAR2(18), BNAME VARCHAR2(18), AMOUNT NUMBER (8,2));


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

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

```
Table created.
```

ii) Insert the data as shown below.

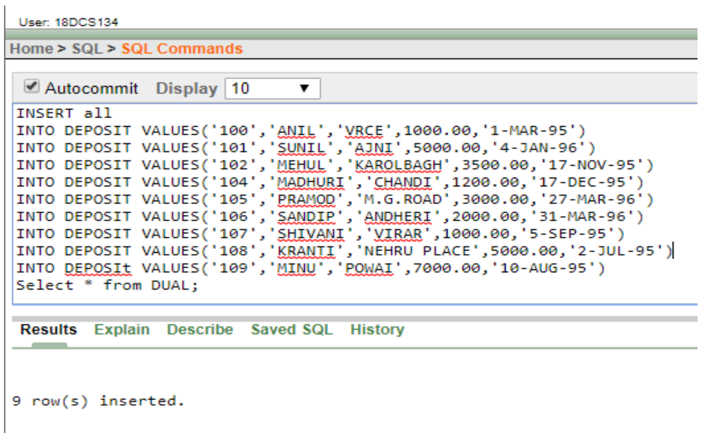


ACTNO	CNAME	BNAME	AMOUNT	ADATE
100	ANIL	VRCE	1000.00	1-MAR-95
101	SUNIL	AJNI	5000.00	4-JAN-96
102	MEHUL	KAROLBAGH	3500.00	17-NOV-95
104	MADHURI	CHANDI	1200.00	17-DEC-95
105	PRMOD	M.G.ROAD	3000.00	27-MAR-96
106	SANDIP	ANDHERI	2000.00	31-MAR-96
107	SHIVANI	VIRAR	1000.00	5-SEP-95
108	KRANTI	NEHRU PLACE	5000.00	2-JUL-95
109	MINU	POWAI	7000.00	10-AUG-95

1.DEPOSIT TABLE:

Query:

```
INSERT all
INTO DEPOSIT VALUES('100','ANIL','VRCE',1000.00,'1-MAR-95')
INTO DEPOSIT VALUES('101','SUNIL','AJNI',5000.00,'4-JAN-96')
INTO DEPOSIT VALUES('102','MEHUL','KAROLBAGH',3500.00,'17-NOV-95')
INTO DEPOSIT VALUES('104','MADHURI','CHANDI',1200.00,'17-DEC-95')
INTO DEPOSIT VALUES('105','PRAMOD','M.G.ROAD',3000.00,'27-MAR-96')
INTO DEPOSIT VALUES('106','SANDIP','ANDHERI',2000.00,'31-MAR-96')
INTO DEPOSIT VALUES('107','SHIVANI','VIRAR',1000.00,'5-SEP-95')
INTO DEPOSIT VALUES('108','KRANTI','NEHRU PLACE',5000.00,'2-JUL-95')
INTO DEPOSIt VALUES('109','MINU','POWAI',7000.00,'10-AUG-95')
Select * from DUAL;
```


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

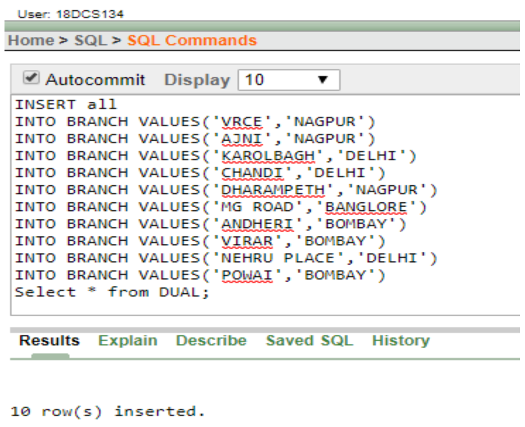
```
INSERT all
INTO DEPOSIT VALUES('100','ANIL','VRCE',1000.00,'1-MAR-95')
INTO DEPOSIT VALUES('101','SUNIL','AJNI',5000.00,'4-JAN-96')
INTO DEPOSIT VALUES('102','MEHUL','KAROLBAGH',3500.00,'17-NOV-95')
INTO DEPOSIT VALUES('104','MADHURI','CHANDI',1200.00,'17-DEC-95')
INTO DEPOSIT VALUES('105','PRAMOD','M.G.ROAD',3000.00,'27-MAR-96')
INTO DEPOSIT VALUES('106','SANDIP','ANDHERI',2000.00,'31-MAR-96')
INTO DEPOSIT VALUES('107','SHIVANI','VIRAR',1000.00,'5-SEP-95')
INTO DEPOSIT VALUES('108','KRANTI','NEHRU PLACE',5000.00,'2-JUL-95')
INTO DEPOSIt VALUES('109','MINU','POWAI',7000.00,'10-AUG-95')
Select * from DUAL;
```

```
9 row(s) inserted.
```

2.Branch Table:

Query:

```
INSERT all
INTO BRANCH VALUES('VRCE','NAGPUR')
INTO BRANCH VALUES('AJNI','NAGPUR')
INTO BRANCH VALUES('KAROLBAGH','DELHI')
INTO BRANCH VALUES('CHANDI','DELHI')
INTO BRANCH VALUES('DHARAMPETH','NAGPUR')
INTO BRANCH VALUES('MG ROAD','BANGLORE')
INTO BRANCH VALUES('ANDHERI','BOMBAY')
INTO BRANCH VALUES('VIRAR','BOMBAY')
INTO BRANCH VALUES('NEHRU PLACE','DELHI')
INTO BRANCH VALUES('POWAI','BOMBAY')
Select * from DUAL;
```


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

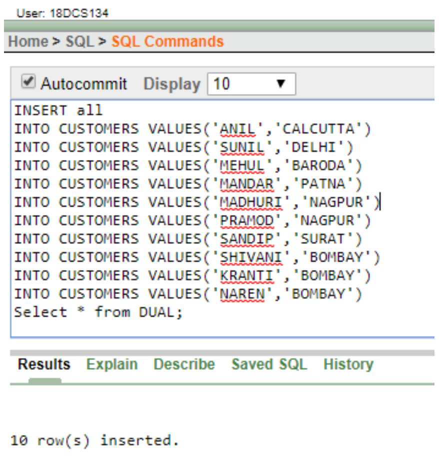
```
INSERT all
INTO BRANCH VALUES('VRCE','NAGPUR')
INTO BRANCH VALUES('AJNI','NAGPUR')
INTO BRANCH VALUES('KAROLBAGH','DELHI')
INTO BRANCH VALUES('CHANDI','DELHI')
INTO BRANCH VALUES('DHARAMPETH','NAGPUR')
INTO BRANCH VALUES('MG ROAD','BANGLORE')
INTO BRANCH VALUES('ANDHERI','BOMBAY')
INTO BRANCH VALUES('VIRAR','BOMBAY')
INTO BRANCH VALUES('NEHRU PLACE','DELHI')
INTO BRANCH VALUES('POWAI','BOMBAY')
Select * from DUAL;
```

```
10 row(s) inserted.
```

3.Customers table:

Query:

```
INSERT all
INTO CUSTOMERS VALUES('ANIL','CALCUTTA')
INTO CUSTOMERS VALUES('SUNIL','DELHI')
INTO CUSTOMERS VALUES('MEHUL','BARODA')
INTO CUSTOMERS VALUES('MANDAR','PATNA')
INTO CUSTOMERS VALUES('MADHURI','NAGPUR')
INTO CUSTOMERS VALUES('PRAMOD','NAGPUR')
INTO CUSTOMERS VALUES('SANDIP','SURAT')
INTO CUSTOMERS VALUES('SHIVANI','BOMBAY')
INTO CUSTOMERS VALUES('KRANTI','BOMBAY')
INTO CUSTOMERS VALUES('NAREN','BOMBAY')
Select * from DUAL;
```



Autocommit Display 10

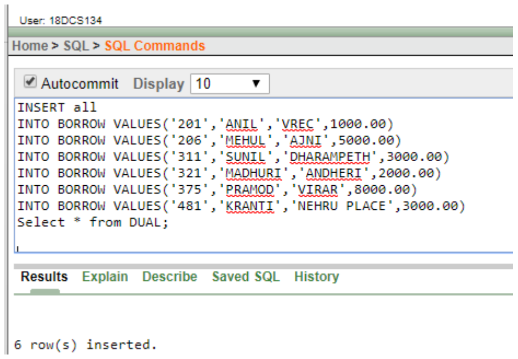
```
INSERT all
INTO CUSTOMERS VALUES('ANIL','CALCUTTA')
INTO CUSTOMERS VALUES('SUNIL','DELHI')
INTO CUSTOMERS VALUES('MEHUL','BARODA')
INTO CUSTOMERS VALUES('MANDAR','PATNA')
INTO CUSTOMERS VALUES('MADHURI','NAGPUR')
INTO CUSTOMERS VALUES('PRAMOD','NAGPUR')
INTO CUSTOMERS VALUES('SANDIP','SURAT')
INTO CUSTOMERS VALUES('SHIVANI','BOMBAY')
INTO CUSTOMERS VALUES('KRANTI','BOMBAY')
INTO CUSTOMERS VALUES('NAREN','BOMBAY')
Select * from DUAL;
```

```
10 row(s) inserted.
```

4.Borrow table:

Query:
INSERT all
INTO BORROW VALUES('201','ANIL','VREC',1000.00)
INTO BORROW VALUES('206','MEHUL','AJNI',5000.00)
INTO BORROW VALUES('311','SUNIL','DHARAMPETH',3000.00)
INTO BORROW VALUES('321','MADHURI','ANDHERI',2000.00)
INTO BORROW VALUES('375','PRAMOD','VIRAR',8000.00)
INTO BORROW VALUES('481','KRANTI','NEHRU PLACE',3000.00)
Select * from DUAL;

Output:



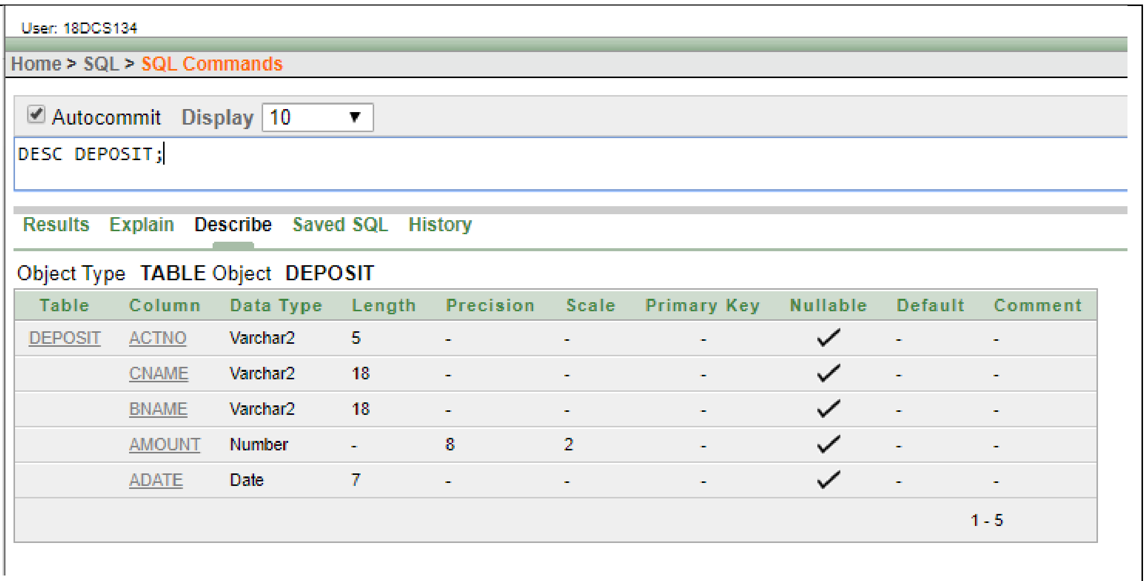
Autocommit Display 10

```
INSERT all
INTO BORROW VALUES('201','ANIL','VREC',1000.00)
INTO BORROW VALUES('206','MEHUL','AJNI',5000.00)
INTO BORROW VALUES('311','SUNIL','DHARAMPETH',3000.00)
INTO BORROW VALUES('321','MADHURI','ANDHERI',2000.00)
INTO BORROW VALUES('375','PRAMOD','VIRAR',8000.00)
INTO BORROW VALUES('481','KRANTI','NEHRU PLACE',3000.00)
Select * from DUAL;
```

```
6 row(s) inserted.
```

From the above given tables perform the following queries:

1) Describe deposit, branch.
Query: DESC DEPOSIT;
OUTPUT:


User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC DEPOSIT;
```

Results Explain Describe Saved SQL History

Object Type TABLE Object DEPOSIT

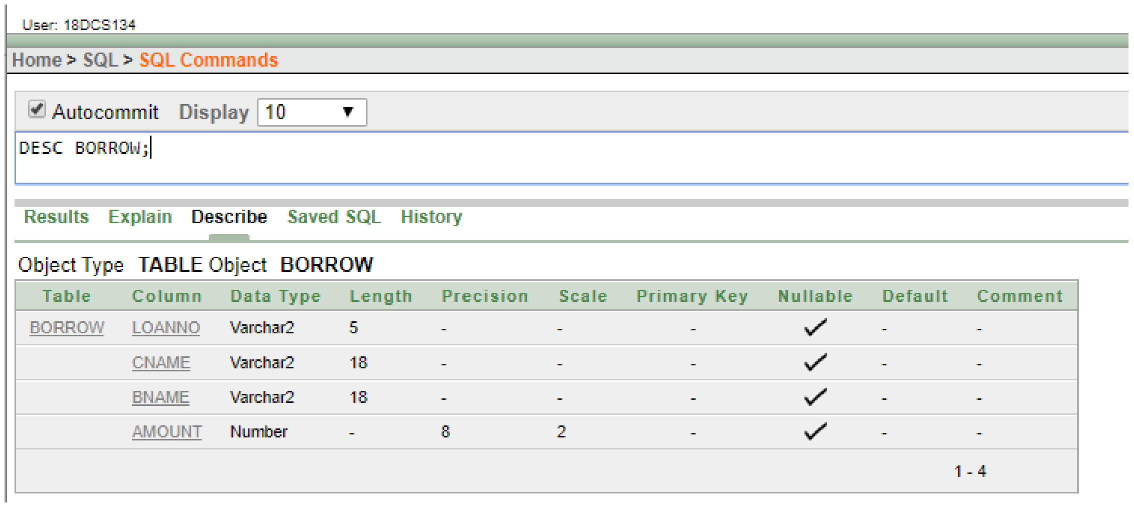
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPOSIT	ACTNO	Varchar2	5	-	-	-	✓	-	-
	CNAME	Varchar2	18	-	-	-	✓	-	-
	BNAME	Varchar2	18	-	-	-	✓	-	-
	AMOUNT	Number	-	8	2	-	✓	-	-
	ADATE	Date	7	-	-	-	✓	-	-

1 - 5

(2) Describe borrow, customers.

<u>Query:</u> DESC BORROW;

<u>Output:</u>


User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC BORROW;
```

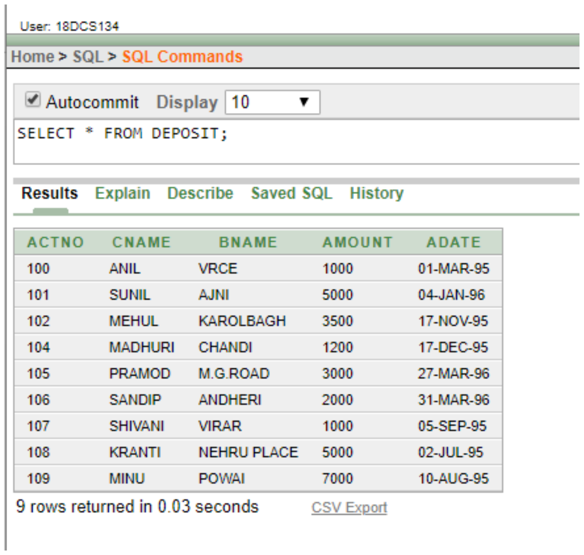
Results Explain Describe Saved SQL History

Object Type TABLE Object BORROW

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
BORROW	LOANNO	Varchar2	5	-	-	-	✓	-	-
	CNAME	Varchar2	18	-	-	-	✓	-	-
	BNAME	Varchar2	18	-	-	-	✓	-	-
	AMOUNT	Number	-	8	2	-	✓	-	-

1 - 4

(3) List all data from table DEPOSIT.
Query: SELECT * FROM DEPOSIT;

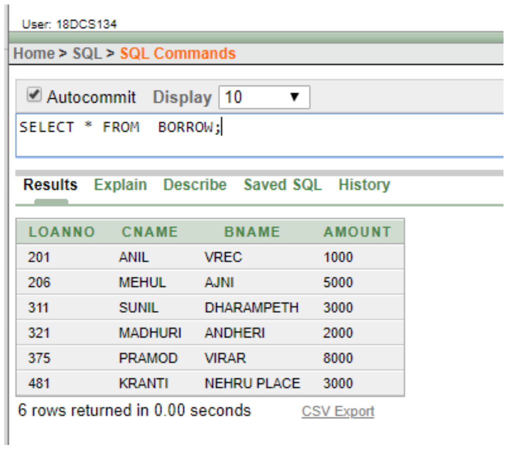

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM DEPOSIT;
```

ACTNO	CNAME	BNAME	AMOUNT	ADATE
100	ANIL	VRCE	1000	01-MAR-95
101	SUNIL	AJNI	5000	04-JAN-96
102	MEHUL	KAROLBAGH	3500	17-NOV-95
104	MADHURI	CHANDI	1200	17-DEC-95
105	PRAMOD	M.G.ROAD	3000	27-MAR-96
106	SANDIP	ANDHERI	2000	31-MAR-96
107	SHIVANI	VIRAR	1000	05-SEP-95
108	KRANTI	NEHRU PLACE	5000	02-JUL-95
109	MINU	POWAI	7000	10-AUG-95

9 rows returned in 0.03 seconds

(4) List all data from table BORROW.
Query: SELECT * FROM BORROW;
Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM  BORROW;
```

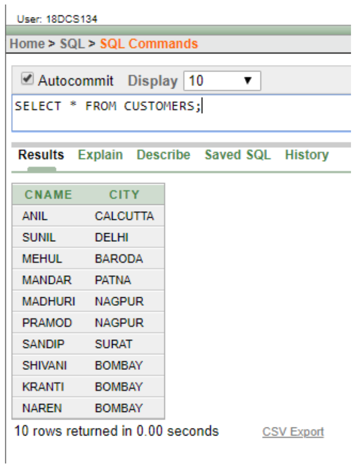
LOANNO	CNAME	BNAME	AMOUNT
201	ANIL	VREC	1000
206	MEHUL	AJNI	5000
311	SUNIL	DHARAMPETH	3000
321	MADHURI	ANDHERI	2000
375	PRAMOD	VIRAR	8000
481	KRANTI	NEHRU PLACE	3000

6 rows returned in 0.00 seconds

(5) List all data from table CUSTOMERS.

Query: SELECT * FROM CUSTOMERS;

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM CUSTOMERS;
```

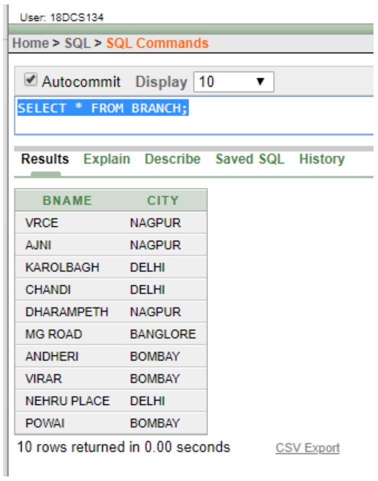
CNAME	CITY
ANIL	CALCUTTA
SUNIL	DELHI
MEHUL	BARODA
MANDAR	PATNA
MADHURI	NAGPUR
PRAMOD	NAGPUR
SANDIP	SURAT
SHIVANI	BOMBAY
KRANTI	BOMBAY
NAREN	BOMBAY

```
10 rows returned in 0.00 seconds
```

(6) List all data from table BRANCH.

Query: SELECT * FROM BRANCH;

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM BRANCH;
```

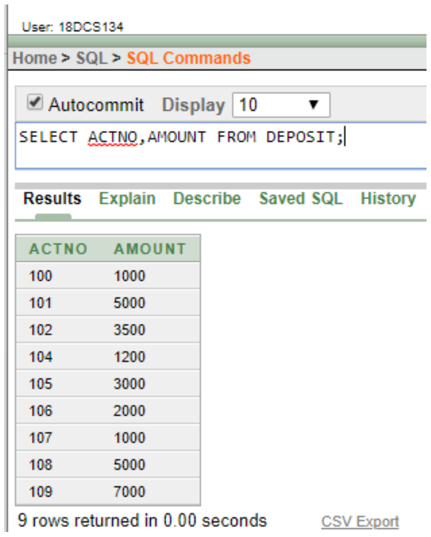
BNAME	CITY
VRCE	NAGPUR
AJNI	NAGPUR
KAROLBAGH	DELHI
CHANDI	DELHI
DHARAMPETH	NAGPUR
MG ROAD	BANGLORE
ANDHERI	BOMBAY
VIRAR	BOMBAY
NEHRU PLACE	DELHI
POWAI	BOMBAY

```
10 rows returned in 0.00 seconds
```

(7) Give account no and amount of depositors.

Query: SELECT ACTNO,AMOUNT FROM DEPOSIT;

Output:


User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT ACTNO,AMOUNT FROM DEPOSIT;
```

Results Explain Describe Saved SQL History

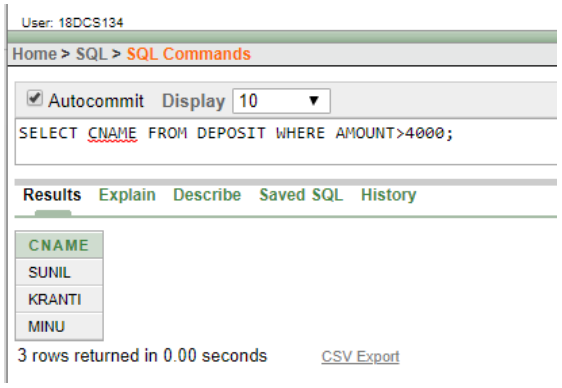
ACTNO	AMOUNT
100	1000
101	5000
102	3500
104	1200
105	3000
106	2000
107	1000
108	5000
109	7000

9 rows returned in 0.00 seconds CSV Export

(8) Give name of depositors having amount greater than 4000.

Query: SELECT CNAME FROM DEPOSIT WHERE AMOUNT>4000;

Output:


User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT CNAME FROM DEPOSIT WHERE AMOUNT>4000;
```

Results Explain Describe Saved SQL History

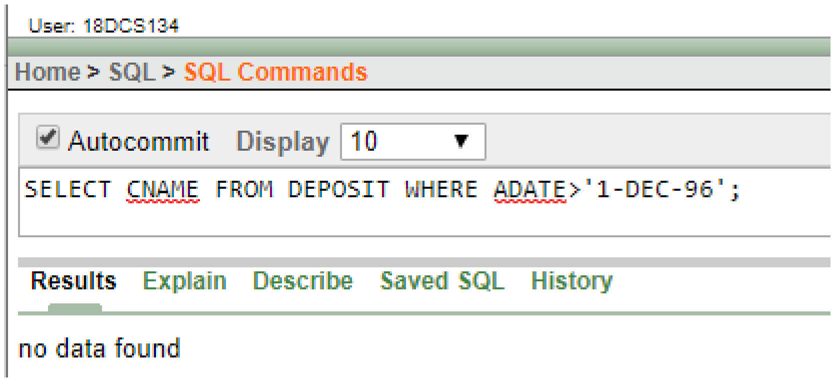
CNAME
SUNIL
KRANTI
MINU

3 rows returned in 0.00 seconds CSV Export

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

Query: SELECT CNAME FROM DEPOSIT WHERE ADATE>'1-DEC-96';

Output:


User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT CNAME FROM DEPOSIT WHERE ADATE>'1-DEC-96';
```

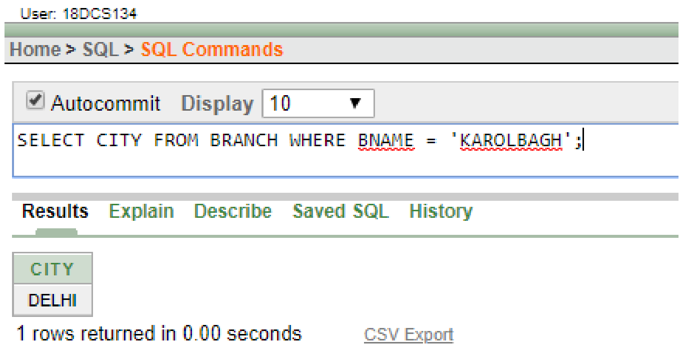
Results Explain Describe Saved SQL History

no data found

(10) Give name of city where branch karolbagh is located.

Query: SELECT CITY FROM BRANCH WHERE BNAME = 'KAROLBAGH';

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT CITY FROM BRANCH WHERE BNAME = 'KAROLBAGH';
```

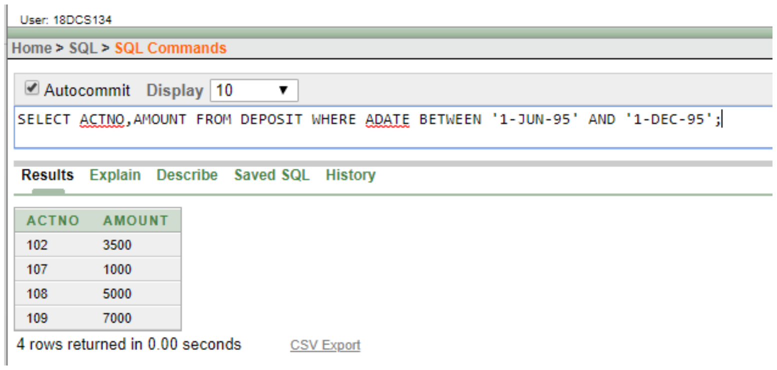
CITY
DELHI

1 rows returned in 0.00 seconds

(11) Give account no and amount of customer having account opened between date 1-
12-95 and 1-6-95.

Query: SELECT ACTNO,AMOUNT FROM DEPOSIT WHERE ADATE BETWEEN '1-JUN-95' AND '1-DEC-95';

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT ACTNO,AMOUNT FROM DEPOSIT WHERE ADATE BETWEEN '1-JUN-95' AND '1-DEC-95';
```

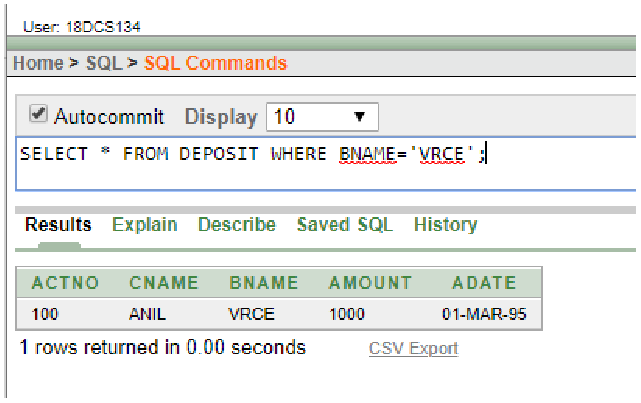
ACTNO	AMOUNT
102	3500
107	1000
108	5000
109	7000

4 rows returned in 0.00 seconds

(12) Give names of depositors having account at VRCE.

Query: SELECT * FROM DEPOSIT WHERE BNAME='VRCE';

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM DEPOSIT WHERE BNAME='VRCE';
```

ACTNO	CNAME	BNAME	AMOUNT	ADATE
100	ANIL	VRCE	1000	01-MAR-95

1 rows returned in 0.00 seconds

Conclusion:

In this practical, we have learned how to create the table, how to insert the data into table and also how to retrieve the data from the table.

3. **Aim:-**
Create the below given table and insert the data accordingly.

Query for Job table:

```
CREATE TABLE JOB(job_id VARCHAR2(15),job_title VARCHAR2(30), min_sal
NUMBER(7,2), max_sal NUMBER(7,2));
```

Insert the data:

```
insert all
into JOB values('IT_PROG','PROGRAMMER','4000','10000')
into JOB values('MK_MGR','MARKETING MANAGER','9000','15000')
into JOB values('FI_MGR','FINANCE MANAGER','8200','12000')
into JOB values('FI_ACC','ACCOUNT','4200','9000')
into JOB values('LEC','LECTURER','6000','17000')
into JOB values('COMP_OP','COMPUTER OPERATOR','1500','3000')
select *from dual;
```

Query for Employee table:

```
CREATE TABLE EMPLOYEE(emp_no NUMBER(3),emp_name VARCHAR2(30),
emp_sal NUMBER(8,2), emp_comm NUMBER(6,1), dept_no NUMBER(3));
```

Insert the data:

```
insert all
into EMPLOYEE values('101','Smith','800','','20')
into EMPLOYEE values('102','Snehal','1600','300','25')
into EMPLOYEE values('103','Adama','1100','0','20')
into EMPLOYEE values('104','Aman','3000','','15')
into EMPLOYEE values('105','Anita','5000','50000','10')
into EMPLOYEE values('106','Sneha','2450','24500','10')
into EMPLOYEE values('107','Anamika','2975','','30')
select *from dual;
```

Query for Deposit table:

```
create table DEPOSIT1(a_no Varchar2(5),cname varchar2(15),bname
varchar2(10),amount number(7,2),a_date date);
```

Insert the data:

```
INSERT ALL
INTO DEPOSIT1 VALUES('101','ANIL','ANDHERI','7000','1-JAN-06')
INTO DEPOSIT1 VALUES('102','SUNIL','VIRAR','5000','15-JUL-06')
```

INTO DEPOSIT1 VALUES('103','JAY','VILLEPARLE','6500','12-MAR-06')
INTO DEPOSIT1 VALUES('104','VIJAY','ANDHERI','8000','17-SEP-06')
INTO DEPOSIT1 VALUES('105','KEYUR','DADAR','7500','19-NOV-06')
INTO DEPOSIT1 VALUES('106','MAYUR','BORIVALI','5500','21-DEC-06')
SELECT *FROM DUAL;

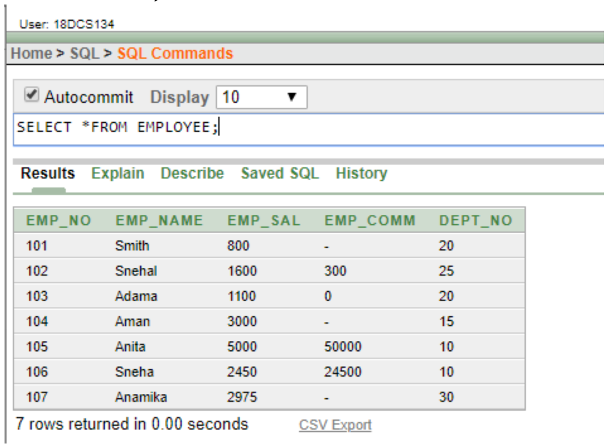
Query for Borrow table:
CREATE TABLE BORROW2(loanno VARCHAR2(5),cname VARCHAR2(15),bname VARCHAR2(10),amount NUMBER(7,2));

Insert the data:
insert all
into borrow2 values(201,'ANIL','VRCE',1000.00)
into borrow2 values(206,'MEHUL','AJNI',5000.00)
into borrow2 values(311,'SUNIL','DHARAMPETH',3000.00)
into borrow2 values(321,'MADHURI','ANDHERI',2000.00)
into borrow2 values(375,'PRMOD','VIRAR',8000.00)
into borrow2 values(481,'KRANTI','NEHRUPLACE',3000.00)
select *from dual;

Perform following queries

1.Retrieve all data from employee, jobs and deposit.

SELECT *FROM EMPLOYEE;

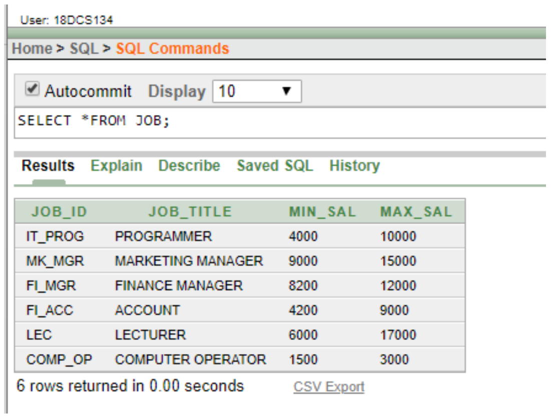

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT *FROM EMPLOYEE;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
101	Smith	800	-	20
102	Snehal	1600	300	25
103	Adama	1100	0	20
104	Aman	3000	-	15
105	Anita	5000	50000	10
106	Sneha	2450	24500	10
107	Anamika	2975	-	30

7 rows returned in 0.00 seconds

Select * from JOB

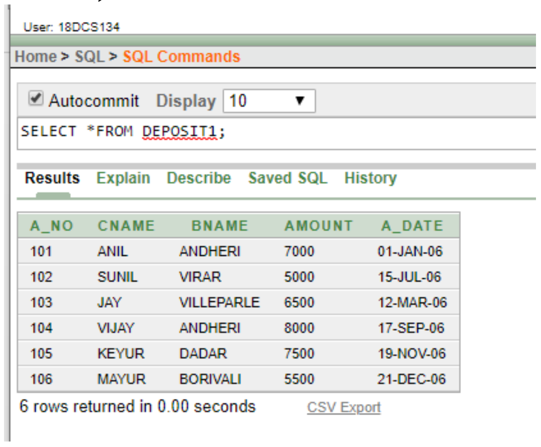

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT *FROM JOB;
```

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
IT_PROG	PROGRAMMER	4000	10000
MK_MGR	MARKETING MANAGER	9000	15000
FI_MGR	FINANCE MANAGER	8200	12000
FI_ACC	ACCOUNT	4200	9000
LEC	LECTURER	6000	17000
COMP_OP	COMPUTER OPERATOR	1500	3000

6 rows returned in 0.00 seconds

SELECT *FROM DEPOSIT1;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

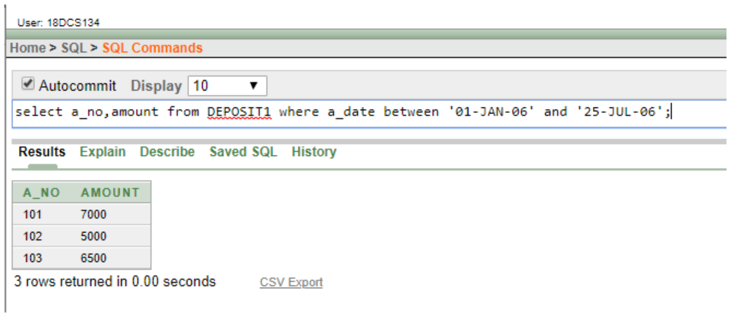
```
SELECT *FROM DEPOSIT1;
```

A_NO	CNAME	BNAME	AMOUNT	A_DATE
101	ANIL	ANDHERI	7000	01-JAN-06
102	SUNIL	VIRAR	5000	15-JUL-06
103	JAY	VILLEPARLE	6500	12-MAR-06
104	VIJAY	ANDHERI	8000	17-SEP-06
105	KEYUR	DADAR	7500	19-NOV-06
106	MAYUR	BORIVALI	5500	21-DEC-06

6 rows returned in 0.00 seconds

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

Query: select a_no,amount from DEPOSIT1 where a_date between '01-JAN-06' and '25-JUL-06';

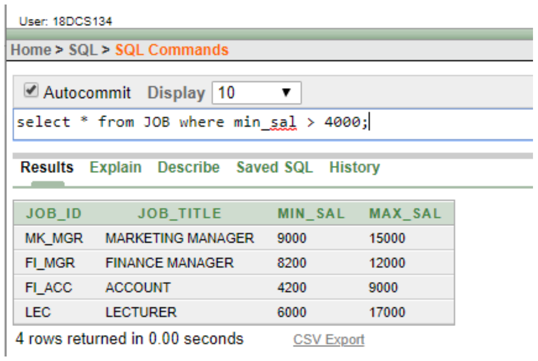

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select a_no,amount from DEPOSIT1 where a_date between '01-JAN-06' and '25-JUL-06';
```

A_NO	AMOUNT
101	7000
102	5000
103	6500

3 rows returned in 0.00 seconds

(3) Display all jobs with minimum salary is greater than 4000.
Query: select * from JOB where min_sal > 4000;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

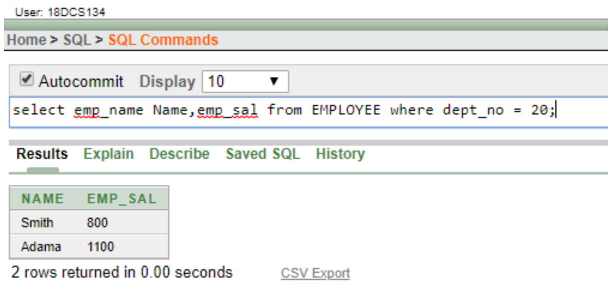
```
select * from JOB where min_sal > 4000;
```

Results Explain Describe Saved SQL History

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
MK_MGR	MARKETING MANAGER	9000	15000
FI_MGR	FINANCE MANAGER	8200	12000
FI_ACC	ACCOUNT	4200	9000
LEC	LECTURER	6000	17000

4 rows returned in 0.00 seconds CSV Export

(4)Display name and salary of employee whose department no is 20. Give alias name to name of employee.
Query: select emp_name Name,emp_sal from EMPLOYEE where dept_no = 20;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

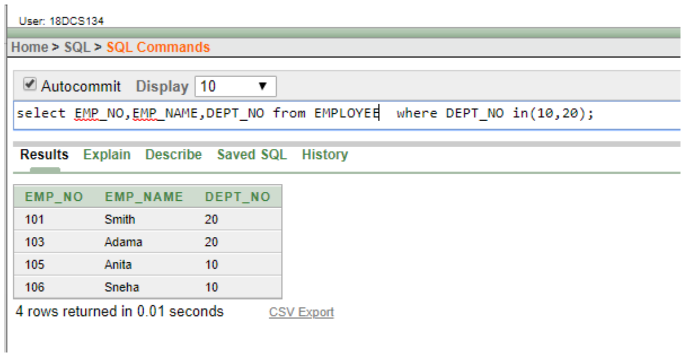
```
select emp_name Name,emp_sal from EMPLOYEE where dept_no = 20;
```

Results Explain Describe Saved SQL History

NAME	EMP_SAL
Smith	800
Adama	1100

2 rows returned in 0.00 seconds CSV Export

(5)Display employee no,name and department details of those employee whose department lies in(10,20).
Query:select EMP_NO,EMP_NAME,DEPT_NO from EMPLOYEE where DEPT_NO in(10,20);


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

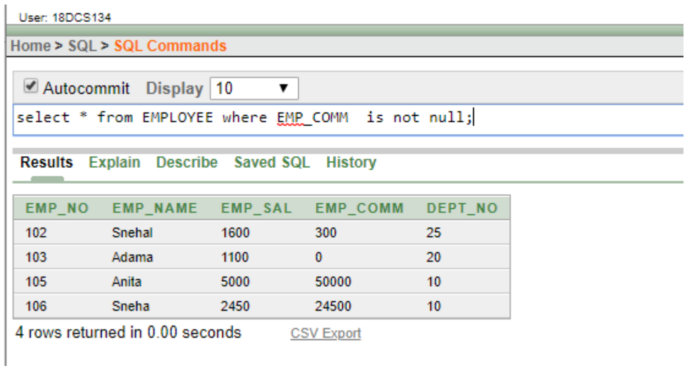
```
select EMP_NO,EMP_NAME,DEPT_NO from EMPLOYEE where DEPT_NO in(10,20);
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	DEPT_NO
101	Smith	20
103	Adama	20
105	Anita	10
106	Sneha	10

4 rows returned in 0.01 seconds CSV Export

(6)Display the non-nullvalues of employees.
Query: select * from EMPLOYEE where EMP_COMM is not null;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

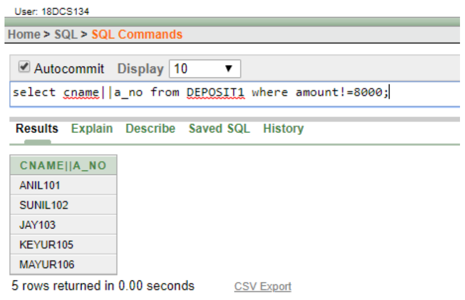
```
select * from EMPLOYEE where EMP_COMM  is not null;
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
102	Snehal	1600	300	25
103	Adama	1100	0	20
105	Anita	5000	50000	10
106	Sneha	2450	24500	10

4 rows returned in 0.00 seconds CSV Export

(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.
Query: select cname||a_no from DEPOSIT1 where amount!=8000;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

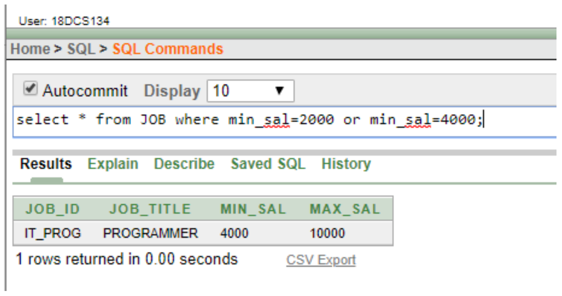
```
select cname||a_no from DEPOSIT1 where amount!=8000;
```

Results Explain Describe Saved SQL History

CNAME||A_NO
ANIL101
SUNIL102
JAY103
KEYUR105
MAYUR106

5 rows returned in 0.00 seconds CSV Export

(8)Display the content of job details with minimum salary either 2000 or 4000.
Query: select * from JOB where min_sal=2000 or min_sal=4000;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

```
select * from JOB where min_sal=2000 or min_sal=4000;
```

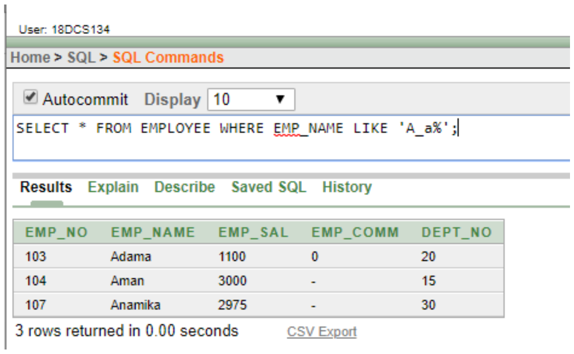
Results Explain Describe Saved SQL History

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
IT_PROG	PROGRAMMER	4000	10000

1 rows returned in 0.00 seconds CSV Export

To study various options of LIKE predicate:

(1)Display all employee whose name start with ‘A’ and third character is ‘ ‘a’.
Query: SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE 'A_a%';

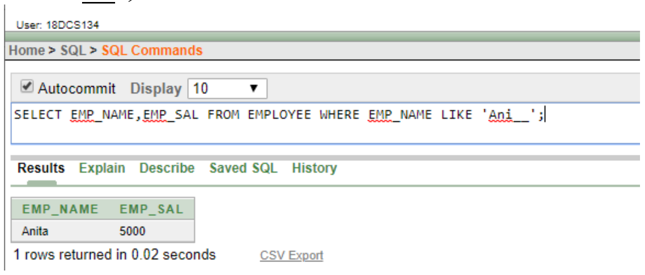

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE 'A_a%';
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
103	Adama	1100	0	20
104	Aman	3000	-	15
107	Anamika	2975	-	30

3 rows returned in 0.00 seconds

(2) Display name, number and salary of those employees whose name is 5 characters long and first three characters are ‘Ani’.
Query: SELECT EMP_NAME,EMP_SAL FROM EMPLOYEE WHERE EMP_NAME LIKE ‘Ani__’;

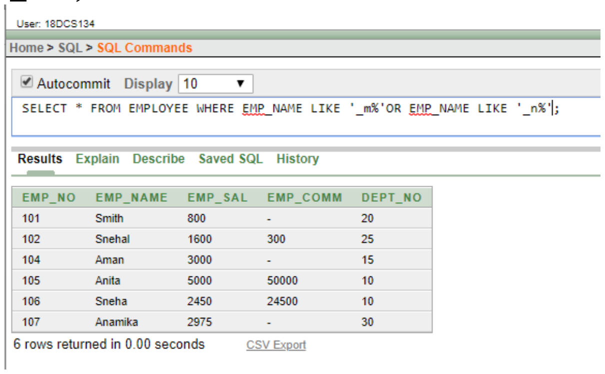

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT EMP_NAME,EMP_SAL FROM EMPLOYEE WHERE EMP_NAME LIKE 'Ani__';
```

EMP_NAME	EMP_SAL
Anita	5000

1 rows returned in 0.02 seconds

(3) Display all information of employee whose second character of name is either ‘M’ or ‘N’.
Query: SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_m%'OR EMP_NAME LIKE '_n%';

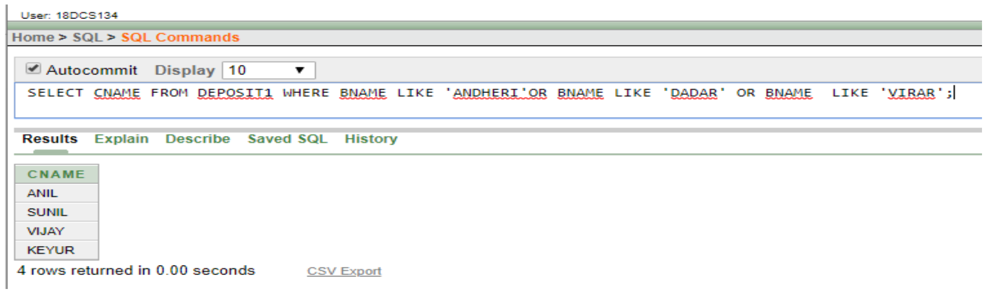

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_m%'OR EMP_NAME LIKE '_n%';
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
101	Smith	800	-	20
102	Snehal	1600	300	25
104	Aman	3000	-	15
105	Anita	5000	50000	10
106	Sneha	2450	24500	10
107	Anamika	2975	-	30

6 rows returned in 0.00 seconds

(4) Find the list of all customer name whose branch is in ‘andheri’ or ‘dadar’ or ‘virar’.
Query: SELECT CNAME FROM DEPOSIT1 WHERE BNAME LIKE 'ANDHERI'OR BNAME LIKE 'DADAR' OR BNAME LIKE 'VIRAR';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

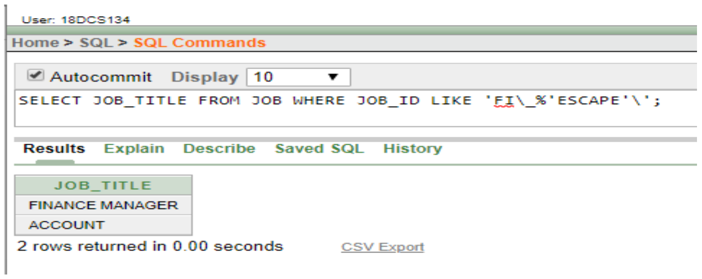
```
SELECT CNAME FROM DEPOSIT1 WHERE BNAME LIKE 'ANDHERI'OR BNAME LIKE 'DADAR' OR BNAME  LIKE 'VIRAR';
```

Results Explain Describe Saved SQL History

CNAME
ANIL
SUNIL
VIJAY
KEYUR

4 rows returned in 0.00 seconds CSV Export

(5)Display the job name whose first three character in job id field is 'FI_'.
Query: SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE 'FI_%'ESCAPE'\';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

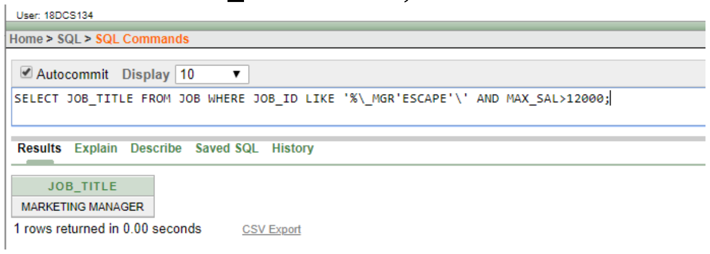
```
SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE 'FI\_%'ESCAPE'\';
```

Results Explain Describe Saved SQL History

JOB_TITLE
FINANCE MANAGER
ACCOUNT

2 rows returned in 0.00 seconds CSV Export

(6)Display the title/name of job whose last three character are '_MGR' and there maximum salary is greater than Rs 12000.
Query: SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE '%_MGR'ESCAPE'\' AND MAX_SAL>12000;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

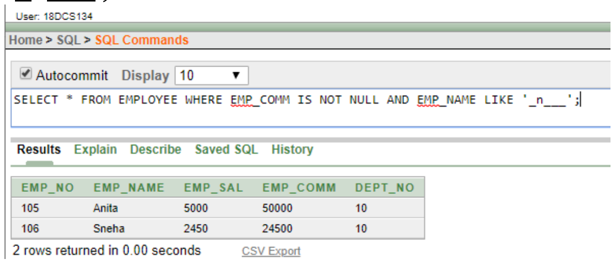
```
SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE '%\_MGR'ESCAPE'\' AND MAX_SAL>12000;
```

Results Explain Describe Saved SQL History

JOB_TITLE
MARKETING MANAGER

1 rows returned in 0.00 seconds CSV Export

(7)Display the non-null values of employees and also employee name second character should be 'n' and string should be 5 character long.
Query: SELECT * FROM EMPLOYEE WHERE EMP_COMM IS NOT NULL AND EMP_NAME LIKE '_n___';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

```
SELECT * FROM EMPLOYEE WHERE EMP_COMM IS NOT NULL AND EMP_NAME LIKE '_n___';
```

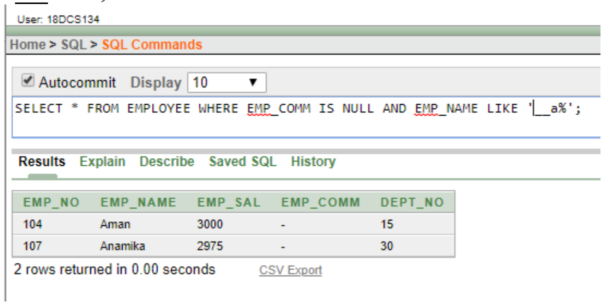
Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
105	Anita	5000	50000	10
106	Sneha	2450	24500	10

2 rows returned in 0.00 seconds CSV Export

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

Query: SELECT * FROM EMPLOYEE WHERE EMP_COMM IS NULL AND EMP_NAME LIKE '__a%';



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM EMPLOYEE WHERE EMP_COMM IS NULL AND EMP_NAME LIKE '__a%';
```

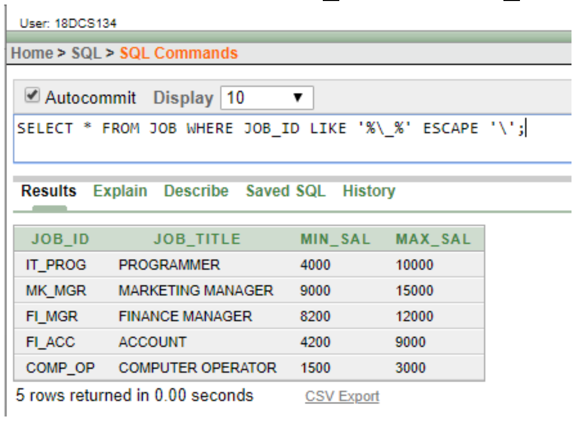
Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
104	Aman	3000	-	15
107	Anamika	2975	-	30

2 rows returned in 0.00 seconds CSV Export

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

Query: SELECT * FROM JOB WHERE JOB_ID LIKE '%_%' ESCAPE '\';



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM JOB WHERE JOB_ID LIKE '%\_%' ESCAPE '\';
```

Results Explain Describe Saved SQL History

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL
IT_PROG	PROGRAMMER	4000	10000
MK_MGR	MARKETING MANAGER	9000	15000
FI_MGR	FINANCE MANAGER	8200	12000
FI_ACC	ACCOUNT	4200	9000
COMP_OP	COMPUTER OPERATOR	1500	3000

5 rows returned in 0.00 seconds CSV Export

Conclusion:

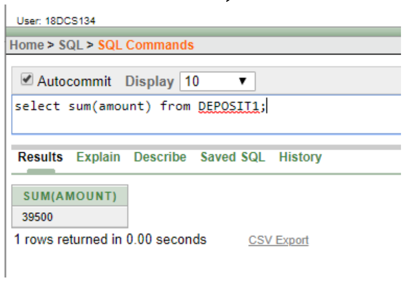
From the above practical,we have studied how to use pipe symbol, how to rename column in output, OR operator and use of LIKE predicate.

4. **Aim:-**

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

(1) List total deposit from deposit.

Query: select sum(amount) from DEPOSIT1;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

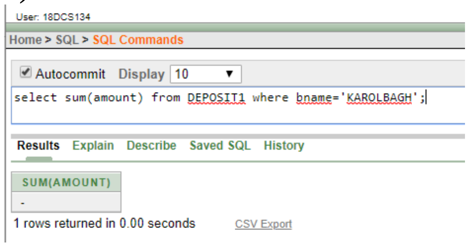
```
select sum(amount) from DEPOSIT1;
```

SUM(AMOUNT)
39500

1 rows returned in 0.00 seconds

(2) List total loan from karolbagh branch

Query: select sum(amount) from DEPOSIT1 where bname='KAROLBAGH';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

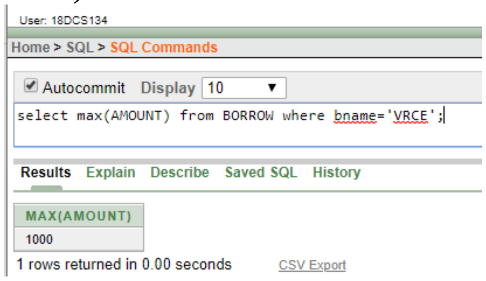
```
select sum(amount) from DEPOSIT1 where bname='KAROLBAGH';
```

SUM(AMOUNT)
-

1 rows returned in 0.00 seconds

(3) Give maximum loan from branch vrce.

Query: select max(AMOUNT) from BORROW where bname='VRCE';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

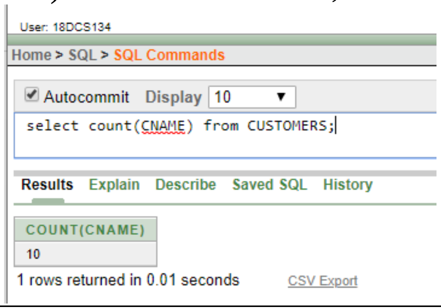
```
select max(AMOUNT) from BORROW where bname='VRCE';
```

MAX(AMOUNT)
1000

1 rows returned in 0.00 seconds

(4) Count total number of customers

Query: select count(CNAME) from CUSTOMERS;

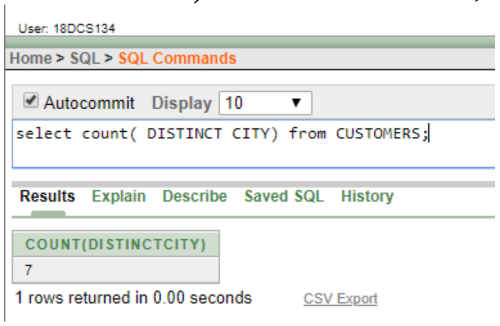

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select count(CNAME) from CUSTOMERS;
```

COUNT(CNAME)
10

1 rows returned in 0.01 seconds

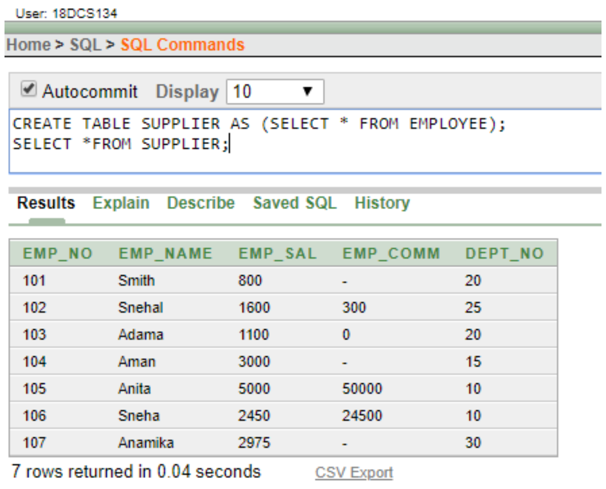
(5) Count total number of customer's cities.
Query: select count(DISTINCT CITY) from CUSTOMERS;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select count( DISTINCT CITY) from CUSTOMERS;
```

```
COUNT(DISTINCTCITY)
7
1 rows returned in 0.00 seconds
```

(6) Create table supplier from employee with all the columns.
Query: CREATE TABLE SUPPLIER AS (SELECT * FROM EMPLOYEE);

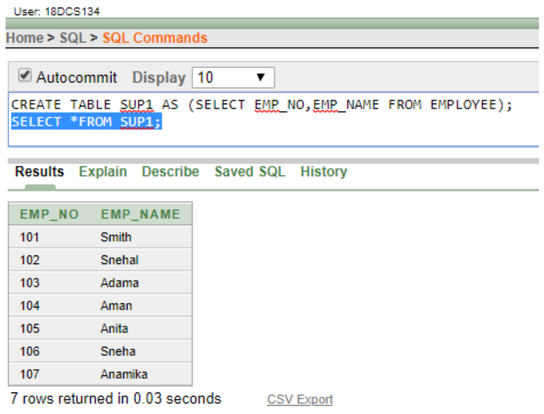

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
CREATE TABLE SUPPLIER AS (SELECT * FROM EMPLOYEE);
SELECT *FROM SUPPLIER;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
101	Smith	800	-	20
102	Snehal	1600	300	25
103	Adama	1100	0	20
104	Aman	3000	-	15
105	Anita	5000	50000	10
106	Sneha	2450	24500	10
107	Anamika	2975	-	30

7 rows returned in 0.04 seconds

(7) Create table sup1 from employee with first two columns.
Query: CREATE TABLE SUP1 AS (SELECT EMP_NO,EMP_NAME FROM EMPLOYEE);


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

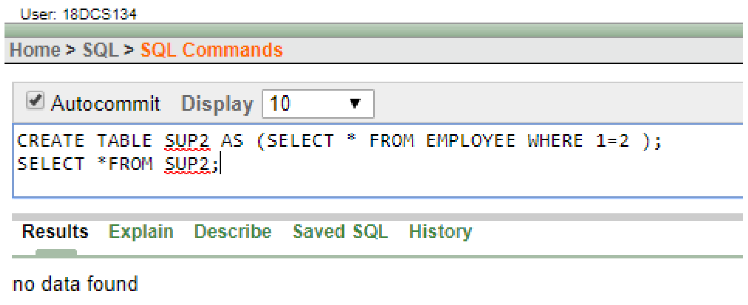
```
CREATE TABLE SUP1 AS (SELECT EMP_NO,EMP_NAME FROM EMPLOYEE);
SELECT *FROM SUP1;
```

EMP_NO	EMP_NAME
101	Smith
102	Snehal
103	Adama
104	Aman
105	Anita
106	Sneha
107	Anamika

7 rows returned in 0.03 seconds

(8) Create table sup2 from employee with no data

Query: CREATE TABLE SUP2 AS (SELECT * FROM EMPLOYEE WHERE 1=2);

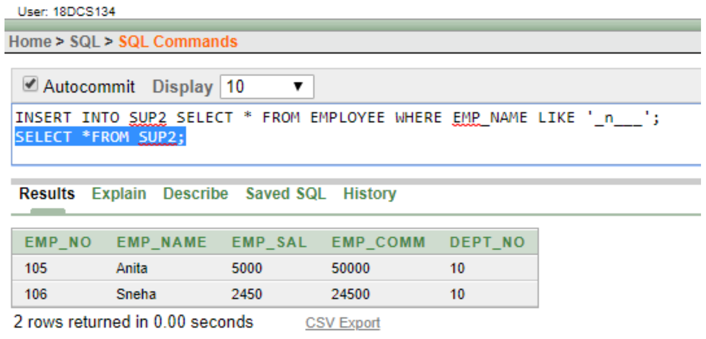

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
CREATE TABLE SUP2 AS (SELECT * FROM EMPLOYEE WHERE 1=2 );
SELECT *FROM SUP2;
```

```
no data found
```

(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.

Query: INSERT INTO SUP2 SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_n___';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

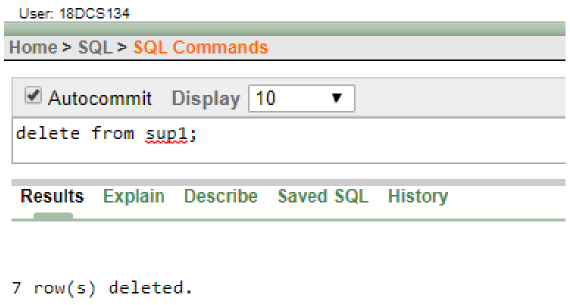
```
INSERT INTO SUP2 SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_n___';
SELECT *FROM SUP2;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
105	Anita	5000	50000	10
106	Sneha	2450	24500	10

2 rows returned in 0.00 seconds

(10) Delete all the rows from sup1.

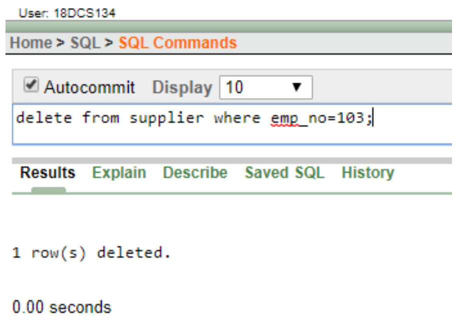
Query: delete from sup1;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
delete from sup1;
```

```
7 row(s) deleted.
```

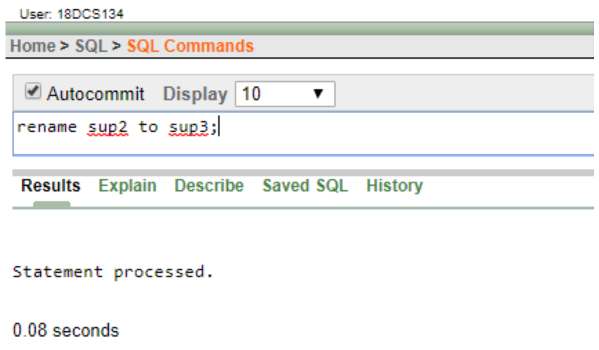
(11) Delete the detail of supplier whose sup_no is 103.
Query: delete from supplier where emp_no=103;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
delete from supplier where emp_no=103;
```

```
1 row(s) deleted.

0.00 seconds
```

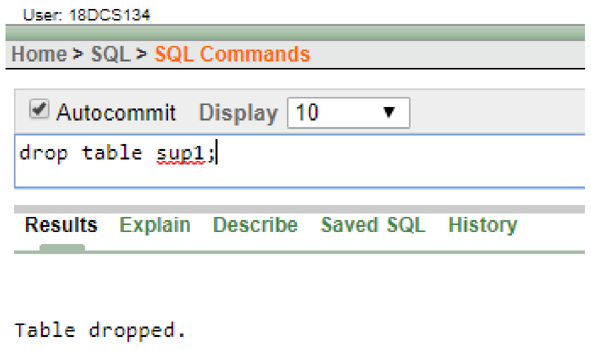
(12) Rename the table sup2.
Query: rename sup2 to sup3;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
rename sup2 to sup3;
```

```
Statement processed.

0.08 seconds
```

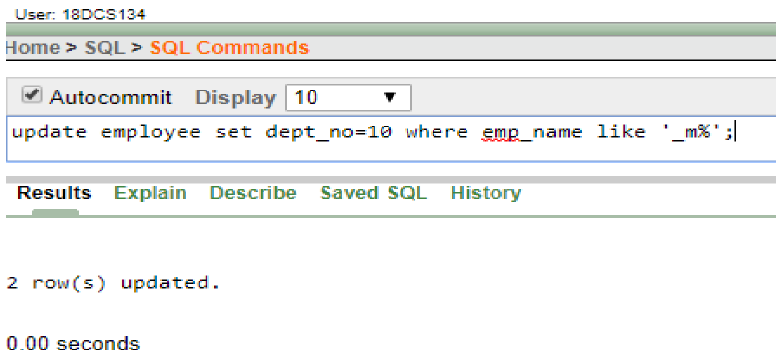
(13) Destroy table sup1 with all the data.
Query: drop table sup1;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
drop table sup1;
```

```
Table dropped.
```

(14) Update the value dept_no to 10 where second character of emp. name is ‘m’.
Query: update employee set dept_no=10 where emp_name like '_m%';

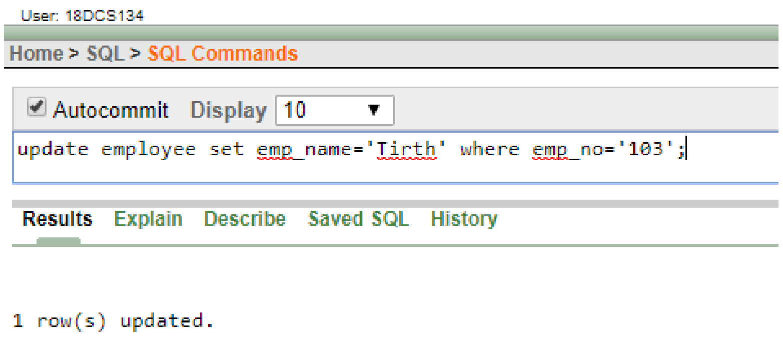

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

```
update employee set dept_no=10 where emp_name like '_m%';
```

Results Explain Describe Saved SQL History

```
2 row(s) updated.

0.00 seconds
```

(15) Update the value of employee name whose employee number is 103.
Query: update employee set emp_name='Tirth' where emp_no='103';

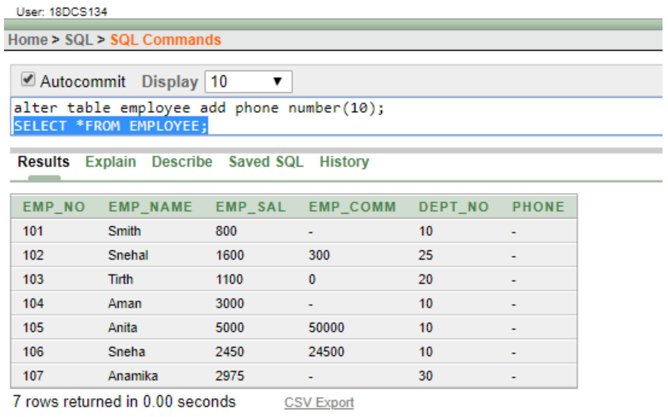

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

```
update employee set emp_name='Tirth' where emp_no='103';
```

Results Explain Describe Saved SQL History

```
1 row(s) updated.
```

(16) Add one column phone to employee with size of column is 10.
Query: alter table employee add phone number(10);


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

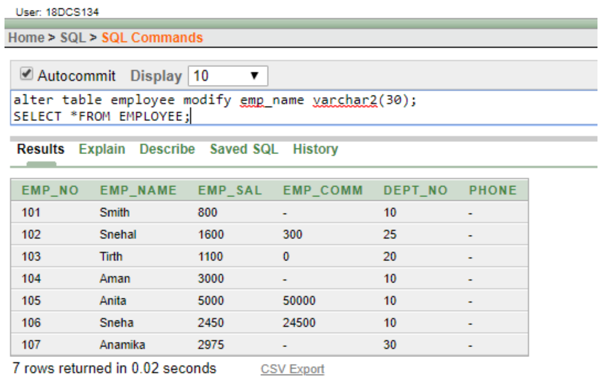
```
alter table employee add phone number(10);
SELECT *FROM EMPLOYEE;
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	PHONE
101	Smith	800	-	10	-
102	Snehal	1600	300	25	-
103	Tirth	1100	0	20	-
104	Aman	3000	-	10	-
105	Anita	5000	50000	10	-
106	Sneha	2450	24500	10	-
107	Anamika	2975	-	30	-

7 rows returned in 0.00 seconds CSV Export

(17) Modify the column emp_name to hold maximum of 30 characters.
Query: alter table employee modify emp_name varchar2(30);



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

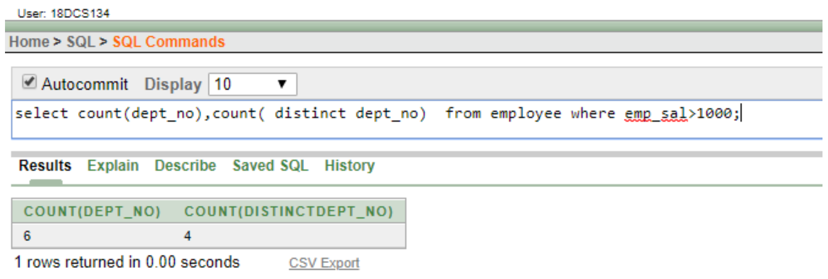
```
alter table employee modify emp_name varchar2(30);
SELECT *FROM EMPLOYEE;
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	PHONE
101	Smith	800	-	10	-
102	Snehal	1600	300	25	-
103	Tirth	1100	0	20	-
104	Aman	3000	-	10	-
105	Anita	5000	50000	10	-
106	Sneha	2450	24500	10	-
107	Anamika	2975	-	30	-

7 rows returned in 0.02 seconds CSV Export

(18) Count the total no as well as distinct rows in dept_no column with a condition of salary greater than 1000 of employee
Query: select count(dept_no),count(distinct dept_no) from employee where emp_sal>1000;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

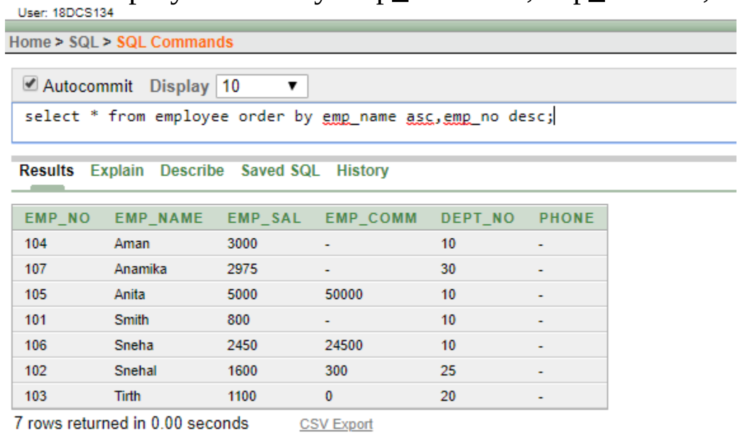
```
select count(dept_no),count( distinct dept_no)  from employee where emp_sal>1000;
```

Results Explain Describe Saved SQL History

COUNT(DEPT_NO)	COUNT(DISTINCTDEPT_NO)
6	4

1 rows returned in 0.00 seconds CSV Export

(19) Display the detail of all employees in ascending order, descending order of their
name and no.
Query: select * from employee order by emp_name asc,emp_no desc;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

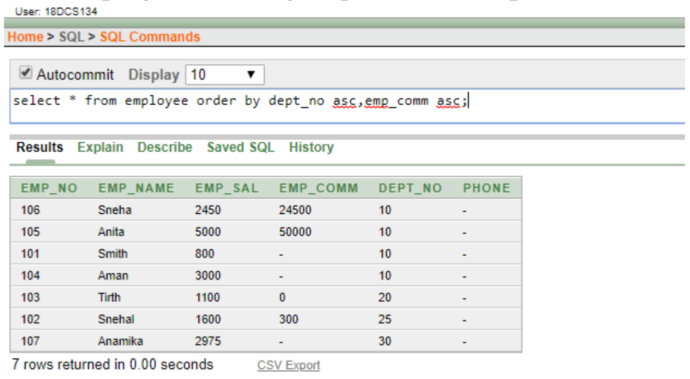
```
select * from employee order by emp_name asc,emp_no desc;
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	PHONE
104	Aman	3000	-	10	-
107	Anamika	2975	-	30	-
105	Anita	5000	50000	10	-
101	Smith	800	-	10	-
106	Sneha	2450	24500	10	-
102	Snehal	1600	300	25	-
103	Tirth	1100	0	20	-

7 rows returned in 0.00 seconds CSV Export

(20) Display the dept_no in ascending order and accordingly display emp_comm in descending order.
Query: select * from employee order by dept_no asc,emp_comm asc;

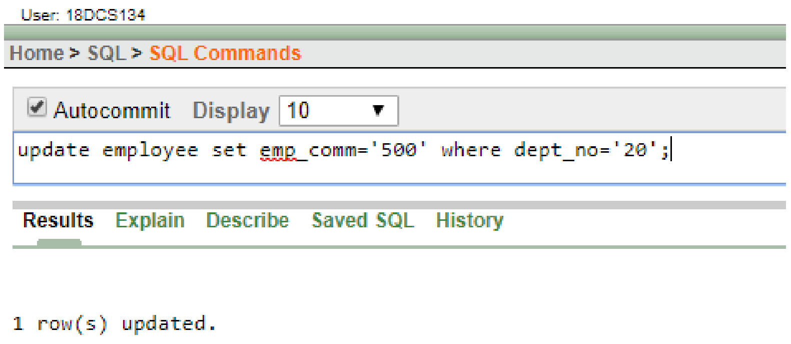

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select * from employee order by dept_no asc,emp_comm asc;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	PHONE
106	Sneha	2450	24500	10	-
105	Anita	5000	50000	10	-
101	Smith	800	-	10	-
104	Aman	3000	-	10	-
103	Tirth	1100	0	20	-
102	Snehal	1600	300	25	-
107	Anamika	2975	-	30	-

7 rows returned in 0.00 seconds

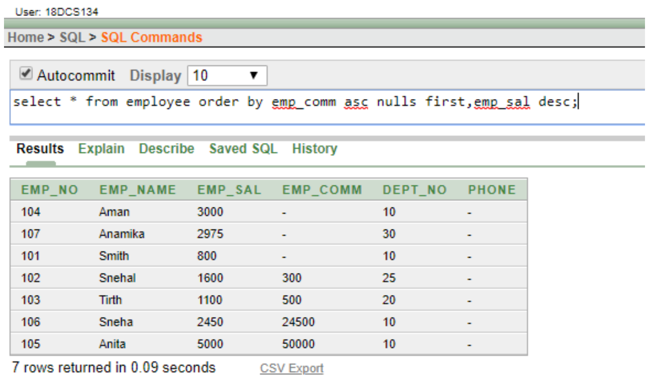
(21) Update the value of emp_comm to 500 where dept_no is 20.
Query: update employee set emp_comm='500' where dept_no='20';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
update employee set emp_comm='500' where dept_no='20';
```

```
1 row(s) updated.
```

(22) Display the emp_comm in ascending order with null value first and accordingly sort employee salary in descending order.
Query: select * from employee order by emp_comm asc nulls first,emp_sal desc;

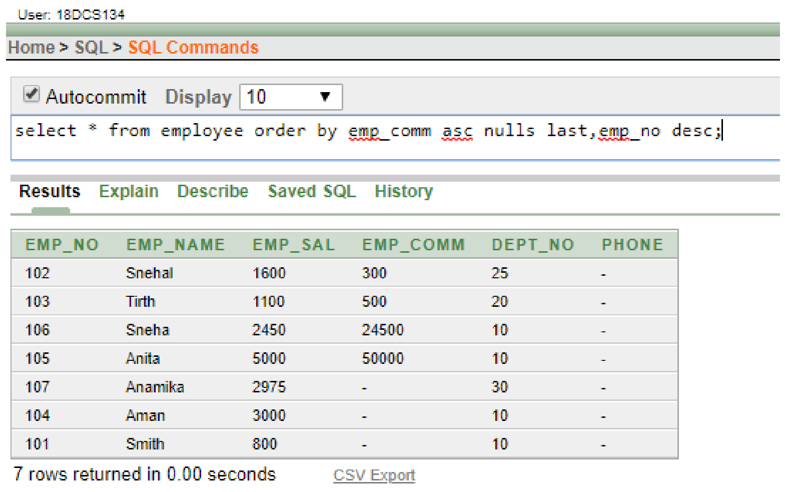

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select * from employee order by emp_comm asc nulls first,emp_sal desc;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	PHONE
104	Aman	3000	-	10	-
107	Anamika	2975	-	30	-
101	Smith	800	-	10	-
102	Snehal	1600	300	25	-
103	Tirth	1100	500	20	-
106	Sneha	2450	24500	10	-
105	Anita	5000	50000	10	-

7 rows returned in 0.09 seconds

(23) Display the emp_comm in ascending order with null value last and accordingly sort emp_no in descending order.
Query: select * from employee order by emp_comm asc nulls last,emp_no desc;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select * from employee order by emp_comm asc nulls last,emp_no desc;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	PHONE
102	Snehal	1600	300	25	-
103	Tirth	1100	500	20	-
106	Sneha	2450	24500	10	-
105	Anita	5000	50000	10	-
107	Anamika	2975	-	30	-
104	Aman	3000	-	10	-
101	Smith	800	-	10	-

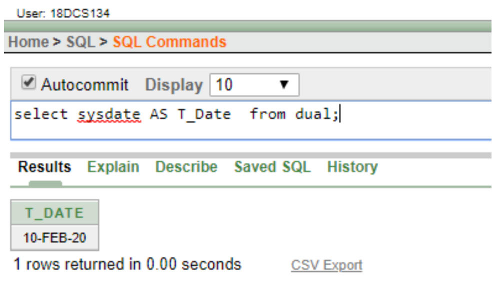
7 rows returned in 0.00 seconds

Conclusion:
From the above practical, we performed various type of aggregate function like count, sum, min, max, etc and also performed sorting operations ascending and descending.

5. **Aim:-To study Single-row functions.**

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

Query: select sysdate AS T_Date from dual;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
select sysdate AS T_Date  from dual;
```

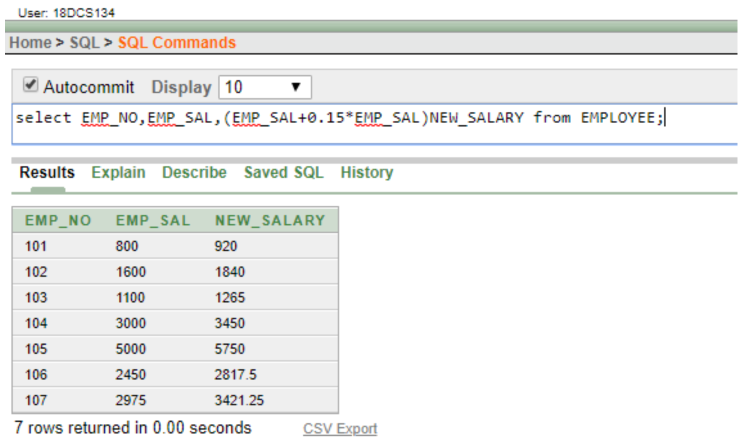
Results Explain Describe Saved SQL History

T_DATE
10-FEB-20

1 rows returned in 0.00 seconds CSV Export

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

Query: select EMP_NO,EMP_SAL,(EMP_SAL+0.15*EMP_SAL)NEW_SALARY from EMPLOYEE;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
select EMP_NO,EMP_SAL,(EMP_SAL+0.15*EMP_SAL)NEW_SALARY from EMPLOYEE;
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_SAL	NEW_SALARY
101	800	920
102	1600	1840
103	1100	1265
104	3000	3450
105	5000	5750
106	2450	2817.5
107	2975	3421.25

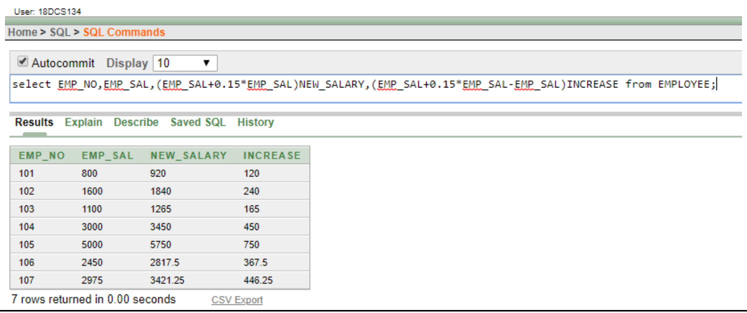
7 rows returned in 0.00 seconds CSV Export

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

Label the column Increase

Query:

select EMP_NO,EMP_SAL,(EMP_SAL+0.15*EMP_SAL)NEW_SALARY,(EMP_SAL+0.15*EMP_SAL-EMP_SAL)INCREASE from EMPLOYEE;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
select EMP_NO,EMP_SAL,(EMP_SAL+0.15*EMP_SAL)NEW_SALARY,(EMP_SAL+0.15*EMP_SAL-EMP_SAL)INCREASE from EMPLOYEE;
```

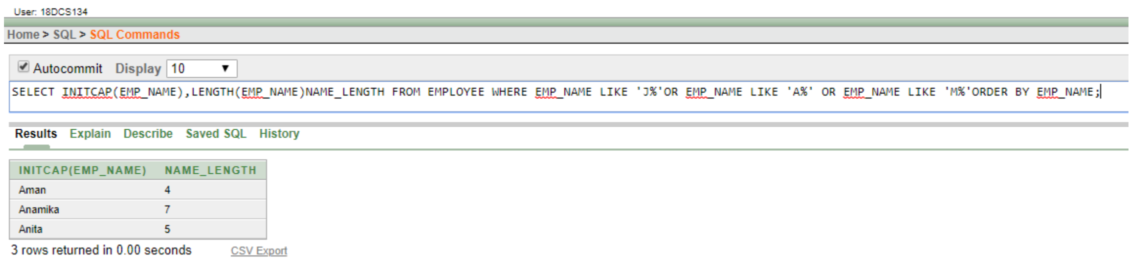
Results Explain Describe Saved SQL History

EMP_NO	EMP_SAL	NEW_SALARY	INCREASE
101	800	920	120
102	1600	1840	240
103	1100	1265	165
104	3000	3450	450
105	5000	5750	750
106	2450	2817.5	367.5
107	2975	3421.25	446.25

7 rows returned in 0.00 seconds CSV Export

(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.

Query: SELECT INITCAP(EMP_NAME),LENGTH(EMP_NAME)NAME_LENGTH FROM EMPLOYEE WHERE EMP_NAME LIKE 'J%'OR EMP_NAME LIKE 'A%' OR EMP_NAME LIKE 'M%'ORDER BY EMP_NAME;

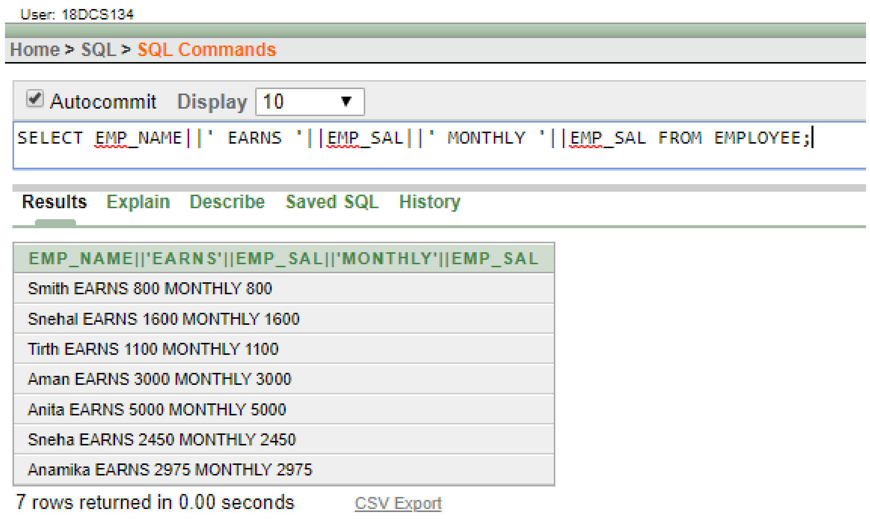

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT INITCAP(EMP_NAME),LENGTH(EMP_NAME)NAME_LENGTH FROM EMPLOYEE WHERE EMP_NAME LIKE 'J%'OR EMP_NAME LIKE 'A%' OR EMP_NAME LIKE 'M%'ORDER BY EMP_NAME;
```

INITCAP(EMP_NAME)	NAME_LENGTH
Aman	4
Anamika	7
Anita	5

3 rows returned in 0.00 seconds

(5) Write a query that produces the following for each employee:
<employee last name> earns <salary> monthly
Query: SELECT EMP_NAME||' EARNS '||EMP_SAL||' MONTHLY '||EMP_SAL FROM EMPLOYEE;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

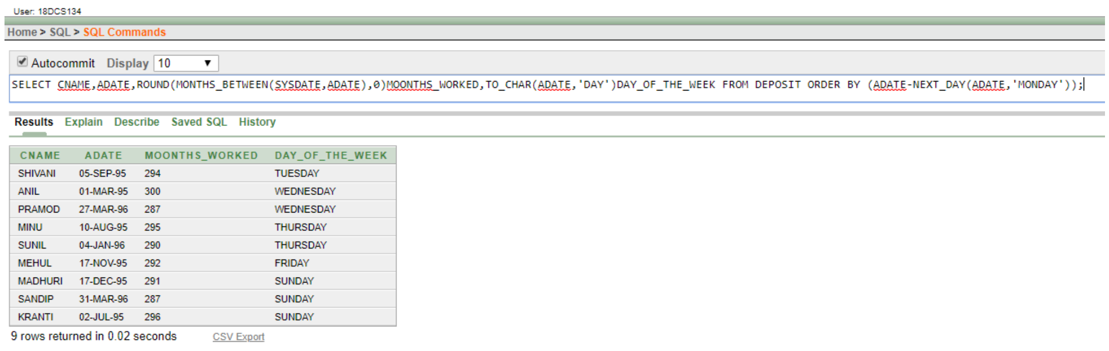
```
SELECT EMP_NAME||' EARNS '||EMP_SAL||' MONTHLY '||EMP_SAL FROM EMPLOYEE;
```

EMP_NAME||'EARNS'||EMP_SAL||'MONTHLY'||EMP_SAL
Smith EARNS 800 MONTHLY 800
Snehal EARNS 1600 MONTHLY 1600
Tirth EARNS 1100 MONTHLY 1100
Aman EARNS 3000 MONTHLY 3000
Anita EARNS 5000 MONTHLY 5000
Sneha EARNS 2450 MONTHLY 2450
Anamika EARNS 2975 MONTHLY 2975

7 rows returned in 0.00 seconds

(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.

Query:SELECT CNAME,ADATE,ROUND(MONTHS_BETWEEN (SYSDATE,ADATE),0)MOONTHS_WORKED,TO_CHAR(ADATE,'DAY')DAY_OF_THE_WEEK FROM DEPOSIT ORDER BY (ADATE-NEXT_DAY(ADATE,'MONDAY'));



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT CNAME,ADATE,ROUND(MONTHS_BETWEEN(SYSDATE,ADATE),0)MOONTHS_WORKED,TO_CHAR(ADATE,'DAY')DAY_OF_THE_WEEK FROM DEPOSIT ORDER BY (ADATE-NEXT_DAY(ADATE,'MONDAY'));
```

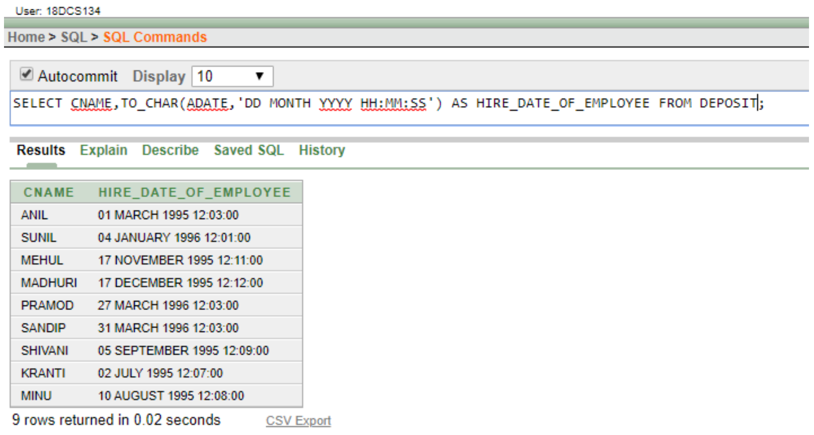
Results Explain Describe Saved SQL History

CNAME	ADATE	MOONTHS_WORKED	DAY_OF_THE_WEEK
SHIVANI	05-SEP-95	294	TUESDAY
ANIL	01-MAR-95	300	WEDNESDAY
PRAMOD	27-MAR-96	287	WEDNESDAY
MINU	10-AUG-95	295	THURSDAY
SUNIL	04-JAN-96	290	THURSDAY
MEHUL	17-NOV-95	292	FRIDAY
MADHURI	17-DEC-95	291	SUNDAY
SANDIP	31-MAR-96	287	SUNDAY
KRANTI	02-JUL-95	296	SUNDAY

9 rows returned in 0.02 seconds CSV Export

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

Query: SELECT CNAME,TO_CHAR(ADATE,'DD MONTH YYYY HH:MM:SS') AS HIRE_DATE_OF_EMPLOYEE FROM DEPOSIT;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

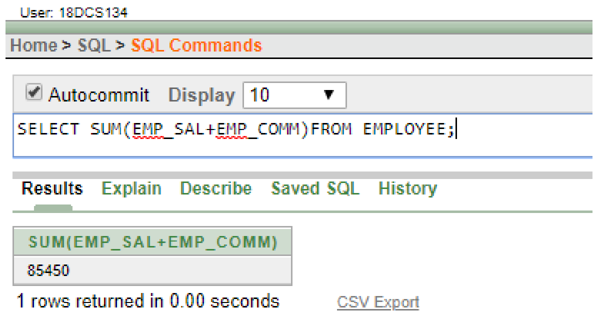
```
SELECT CNAME,TO_CHAR(ADATE,'DD MONTH YYYY HH:MM:SS') AS HIRE_DATE_OF_EMPLOYEE FROM DEPOSIT;
```

Results Explain Describe Saved SQL History

CNAME	HIRE_DATE_OF_EMPLOYEE
ANIL	01 MARCH 1995 12:03:00
SUNIL	04 JANUARY 1996 12:01:00
MEHUL	17 NOVEMBER 1995 12:11:00
MADHURI	17 DECEMBER 1995 12:12:00
PRAMOD	27 MARCH 1996 12:03:00
SANDIP	31 MARCH 1996 12:03:00
SHIVANI	05 SEPTEMBER 1995 12:09:00
KRANTI	02 JULY 1995 12:07:00
MINU	10 AUGUST 1995 12:08:00

9 rows returned in 0.02 seconds CSV Export

(8) Write a query to calculate the annual compensation of all employees (sal+comm.).
Query: SELECT SUM(EMP_SAL+EMP_COMM)FROM EMPLOYEE;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT SUM(EMP_SAL+EMP_COMM)FROM EMPLOYEE;
```

```
SUM(EMP_SAL+EMP_COMM)
85450
1 rows returned in 0.00 seconds
```

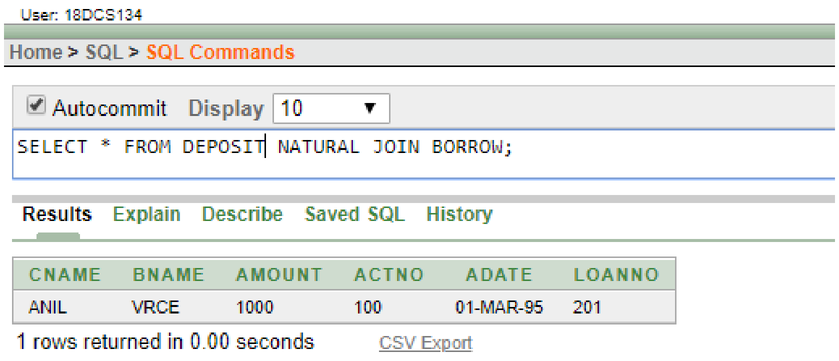
<u>**Conclusion:**</u>
In this Practicala,we Performed single row functions.

6. **Aim:-**

Displaying data from Multiple Tables (join)

(1) Give details of customers ANIL.

Query: SELECT * FROM DEPOSIT NATURAL JOIN BORROW;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

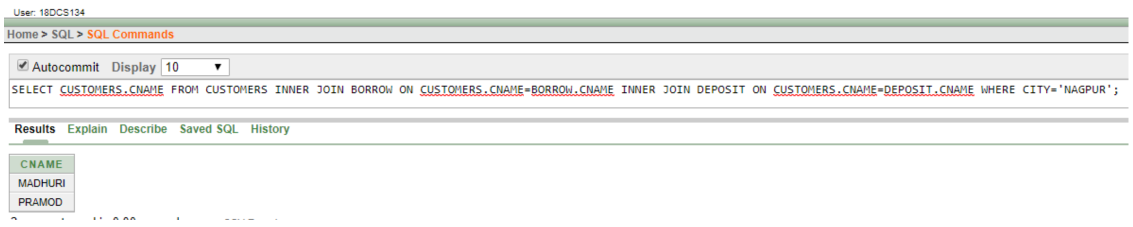
```
SELECT * FROM DEPOSIT NATURAL JOIN BORROW;
```

CNAME	BNAME	AMOUNT	ACTNO	ADATE	LOANNO
ANIL	VRCE	1000	100	01-MAR-95	201

1 rows returned in 0.00 seconds

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

Query: SELECT CUSTOMERS.CNAME FROM CUSTOMERS INNER JOIN BORROW ON CUSTOMERS.CNAME=BORROW.CNAME INNER JOIN DEPOSIT2 ON CUSTOMERS.CNAME=DEPOSIT2.CNAME WHERE CITY='NAGPUR';

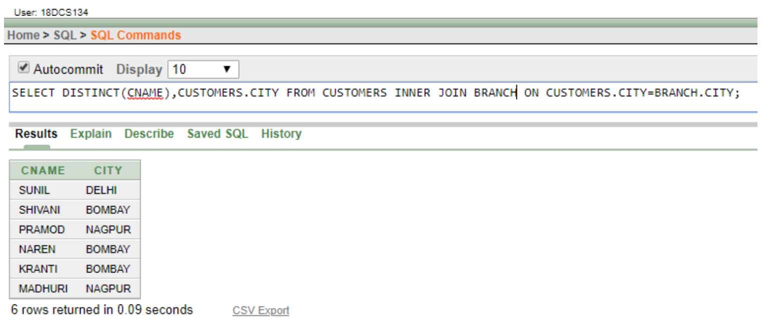

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
SELECT CUSTOMERS.CNAME FROM CUSTOMERS INNER JOIN BORROW ON CUSTOMERS.CNAME=BORROW.CNAME INNER JOIN DEPOSIT ON CUSTOMERS.CNAME=DEPOSIT.CNAME WHERE CITY='NAGPUR';
```

CNAME
MADHURI
PRAMOD

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

Query: SELECT DISTINCT(CNAME),CUSTOMERS.CITY FROM CUSTOMERS INNER JOIN BRANCH1 ON CUSTOMERS.CITY=BRANCH1.CITY;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

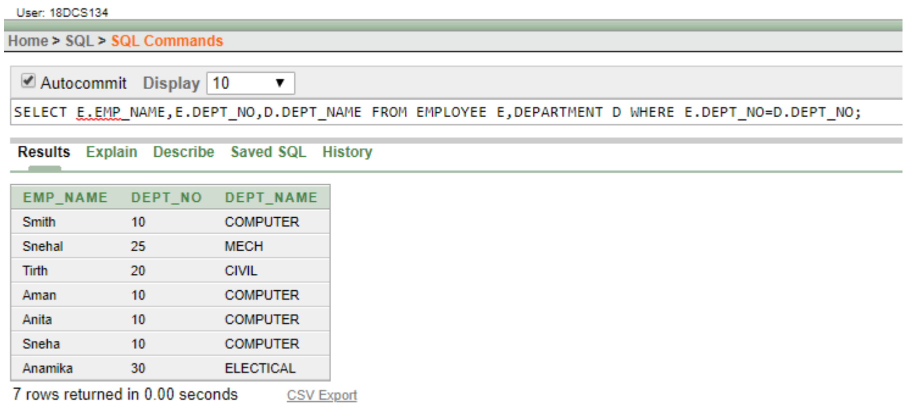
```
SELECT DISTINCT(CNAME),CUSTOMERS.CITY FROM CUSTOMERS INNER JOIN BRANCH ON CUSTOMERS.CITY=BRANCH.CITY;
```

CNAME	CITY
SUNIL	DELHI
SHIVANI	BOMBAY
PRAMOD	NAGPUR
NAREN	BOMBAY
KRANTI	BOMBAY
MADHURI	NAGPUR

6 rows returned in 0.09 seconds

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

Query: SELECT E.EMP_NAME,E.DEPT_NO,D.DEPT_NAME FROM EMPLOYEE E,DEPARTMENT D WHERE E.DEPT_NO=D.DEPT_NO;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT E.EMP_NAME,E.DEPT_NO,D.DEPT_NAME FROM EMPLOYEE E,DEPARTMENT D WHERE E.DEPT_NO=D.DEPT_NO;
```

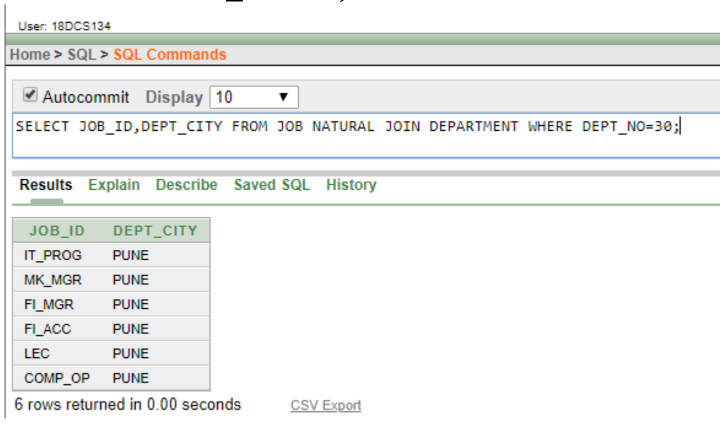
Results Explain Describe Saved SQL History

EMP_NAME	DEPT_NO	DEPT_NAME
Smith	10	COMPUTER
Snehal	25	MECH
Tirth	20	CIVIL
Aman	10	COMPUTER
Anita	10	COMPUTER
Sneha	10	COMPUTER
Anamika	30	ELECTICAL

7 rows returned in 0.00 seconds CSV Export

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

Query: SELECT JOB_ID,DEPT_LOC FROM JOB NATURAL JOIN DEPARTMENT WHERE DEPT_NO=30;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT JOB_ID,DEPT_CITY FROM JOB NATURAL JOIN DEPARTMENT WHERE DEPT_NO=30;
```

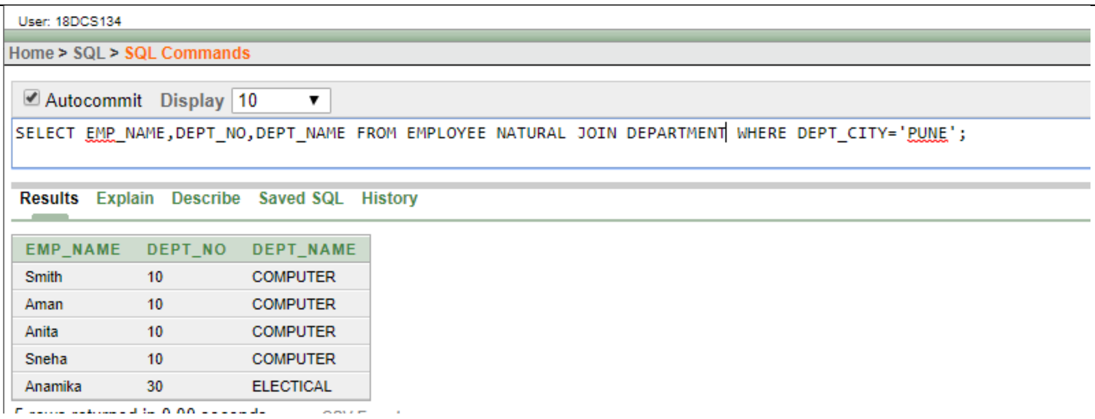
Results Explain Describe Saved SQL History

JOB_ID	DEPT_CITY
IT_PROG	PUNE
MK_MGR	PUNE
FI_MGR	PUNE
FI_ACC	PUNE
LEC	PUNE
COMP_OP	PUNE

6 rows returned in 0.00 seconds CSV Export

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

Query: SELECT EMP_NAME,DEPT_NO,DEPT_NAME FROM EMPLOYEE NATURAL JOIN JOB WHERE DEPT_LOC='DAKOR';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
SELECT EMP_NAME,DEPT_NO,DEPT_NAME FROM EMPLOYEE NATURAL JOIN DEPARTMENT WHERE DEPT_CITY='PUNE';
```

EMP_NAME	DEPT_NO	DEPT_NAME
Smith	10	COMPUTER
Aman	10	COMPUTER
Anita	10	COMPUTER
Sneha	10	COMPUTER
Anamika	30	ELECTICAL

(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.
Query: SELECT EMP_NAME,(EMP_NO)EMP#,MAN_NAME,(MAN_NO)MAN# FROM EMPLOYEE NATURAL JOIN MANAGER;

(8) Create a query to display the name and hire date of any employee hired after employee Sneha.
Query: SELECT EMP_NAME,HIRE_DATE FROM EMPLOYEE WHERE HIRE_DATE>(SELECT HIRE_DATE FROM EMPLOYEE WHERE EMP_NAME='Sneha');

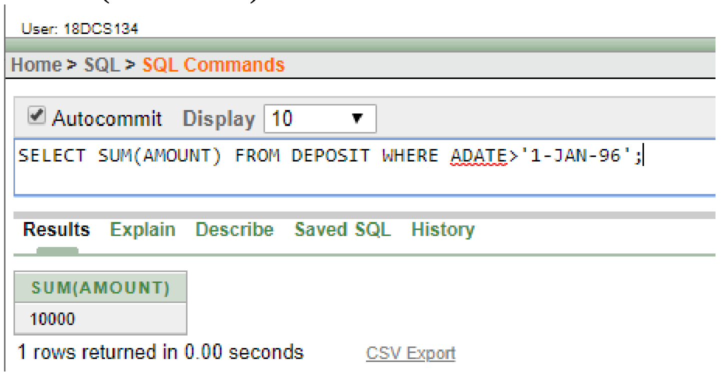
Conclusion : From the above practical,we have studied how to get information from combining 2 or more than two table.

7. **Aim:-**

To apply the concept of Aggregating Data using Group functions.

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

Query: SELECT SUM(AMOUNT) FROM DEPOSIT WHERE ADATE>'1-JAN-96';

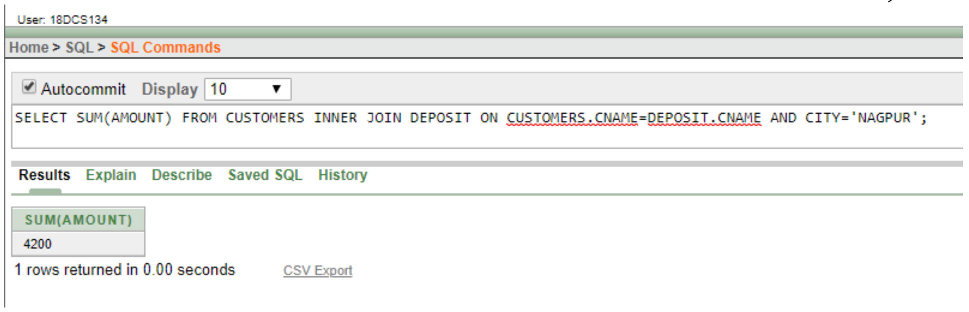

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT SUM(AMOUNT) FROM DEPOSIT WHERE ADATE>'1-JAN-96';
```

```
SUM(AMOUNT)
10000
1 rows returned in 0.00 seconds
```

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

Query: SELECT SUM(AMOUNT) FROM CUSTOMERS INNER JOIN DEPOSIT ON CUSTOMERS.CNAME=DEPOSIT.CNAME AND CITY='NAGPUR';

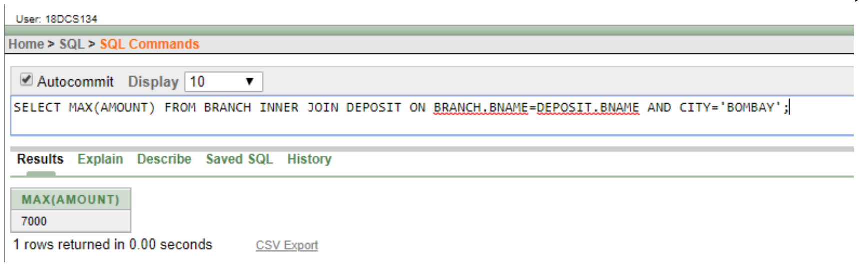

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT SUM(AMOUNT) FROM CUSTOMERS INNER JOIN DEPOSIT ON CUSTOMERS.CNAME=DEPOSIT.CNAME AND CITY='NAGPUR';
```

```
SUM(AMOUNT)
4200
1 rows returned in 0.00 seconds
```

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

Query: SELECT MAX(AMOUNT) FROM BRANCH INNER JOIN DEPOSIT ON BRANCH.BNAME=DEPOSIT.BNAME AND CITY='BOMBAY';

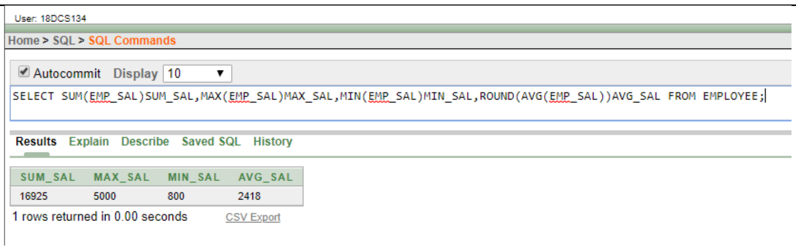

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT MAX(AMOUNT) FROM BRANCH INNER JOIN DEPOSIT ON BRANCH.BNAME=DEPOSIT.BNAME AND CITY='BOMBAY';
```

```
MAX(AMOUNT)
7000
1 rows returned in 0.00 seconds
```

(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.

Query: SELECT SUM(EMP_SAL)SUM_SAL,MAX(EMP_SAL)MAX_SAL,MIN(EMP_SAL)MIN_SAL,ROUND(AVG(EMP_SAL))AVG_SAL FROM EMPLOYEE;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

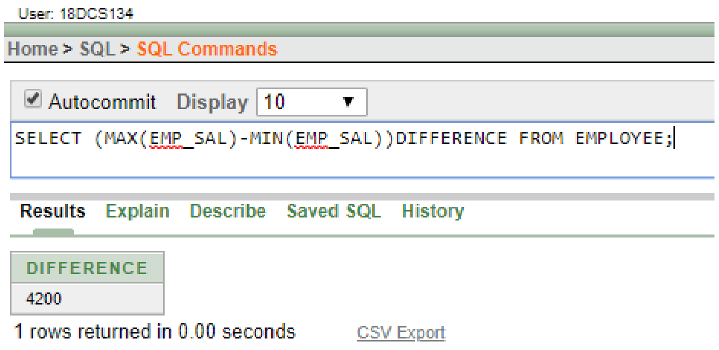
```
SELECT SUM(EMP_SAL)SUM_SAL,MAX(EMP_SAL)MAX_SAL,MIN(EMP_SAL)MIN_SAL,ROUND(AVG(EMP_SAL))AVG_SAL FROM EMPLOYEE;
```

Results Explain Describe Saved SQL History

SUM_SAL	MAX_SAL	MIN_SAL	AVG_SAL
16925	5000	800	2418

1 rows returned in 0.00 seconds CSV Export

(5) Write a query that displays the difference between the highest and lowest salaries. Label the column DIFFERENCE.
Query: SELECT (MAX(EMP_SAL)-MIN(EMP_SAL))DIFFERENCE FROM EMPLOYEE;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

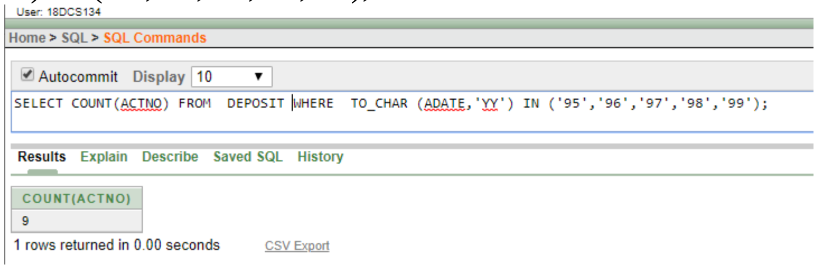
```
SELECT (MAX(EMP_SAL)-MIN(EMP_SAL))DIFFERENCE FROM EMPLOYEE;
```

Results Explain Describe Saved SQL History

DIFFERENCE
4200

1 rows returned in 0.00 seconds CSV Export

(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
Query: SELECT COUNT(ACTNO) FROM DEPOSIT WHERE TO_CHAR (ADATE,'YY') IN ('95','96','97','98','99');



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

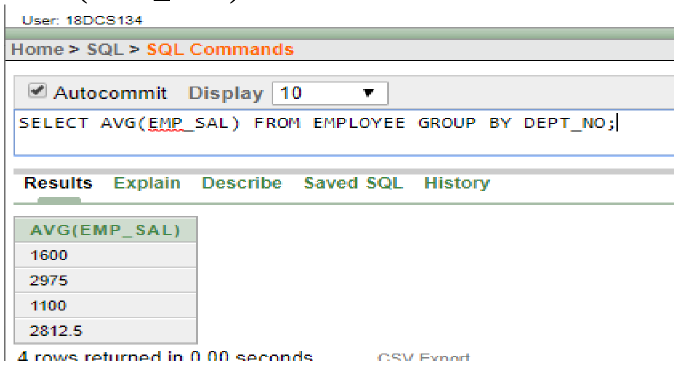
```
SELECT COUNT(ACTNO) FROM  DEPOSIT WHERE  TO_CHAR (ADATE,'YY') IN ('95','96','97','98','99');
```

Results Explain Describe Saved SQL History

COUNT(ACTNO)
9

1 rows returned in 0.00 seconds CSV Export

(7) Find the average salaries for each department without displaying the respective department numbers.
Query: SELECT AVG(EMP_SAL) FROM EMPLOYEE GROUP BY DEPT_NO;

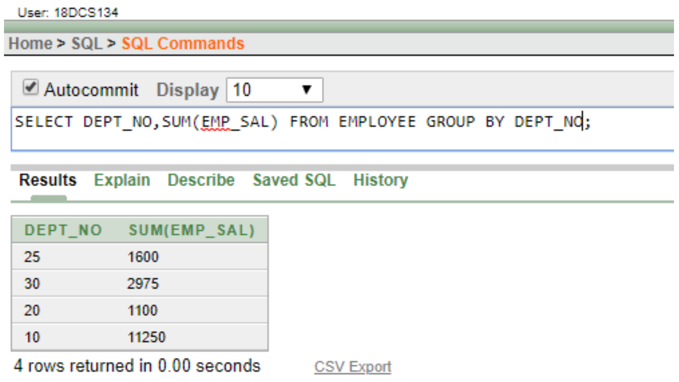

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT AVG(EMP_SAL) FROM EMPLOYEE GROUP BY DEPT_NO;
```

AVG(EMP_SAL)
1600
2975
1100
2812.5

4 rows returned in 0.00 seconds

(8) Write a query to display the total salary being paid to each job title, within each department.
Query: SELECT DEPT_NO,SUM(EMP_SAL) FROM EMPLOYEE GROUP BY DEPT_NO;

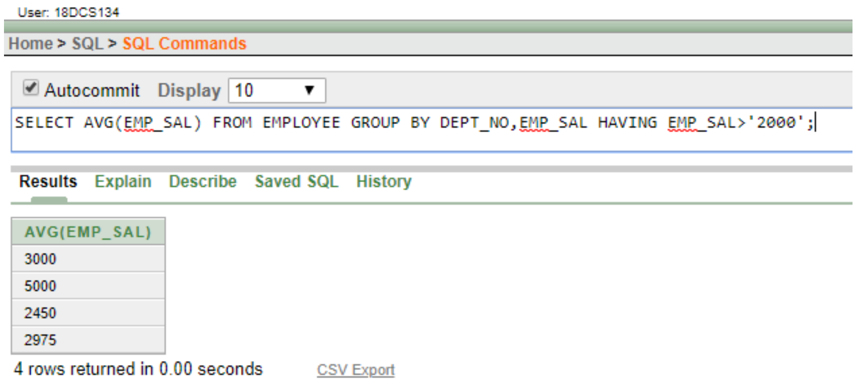

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT DEPT_NO,SUM(EMP_SAL) FROM EMPLOYEE GROUP BY DEPT_NO;
```

DEPT_NO	SUM(EMP_SAL)
25	1600
30	2975
20	1100
10	11250

4 rows returned in 0.00 seconds

(9) Find the average salaries > 2000 for each department without displaying the respective department numbers.
Query: SELECT AVG(EMP_SAL) FROM EMPLOYEE GROUP BY DEPT_NO,EMP_SAL HAVING EMP_SAL>'2000';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

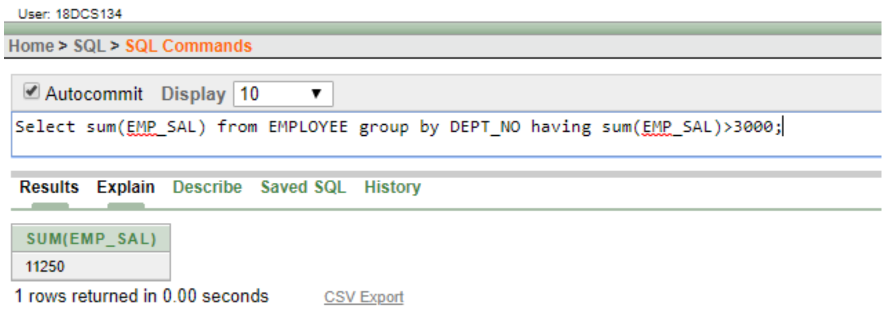
```
SELECT AVG(EMP_SAL) FROM EMPLOYEE GROUP BY DEPT_NO,EMP_SAL HAVING EMP_SAL>'2000';
```

AVG(EMP_SAL)
3000
5000
2450
2975

4 rows returned in 0.00 seconds

(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.

Query: Select sum(EMP_SAL) from EMPLOYEE group by DEPT_NO having sum(EMP_SAL)>3000;

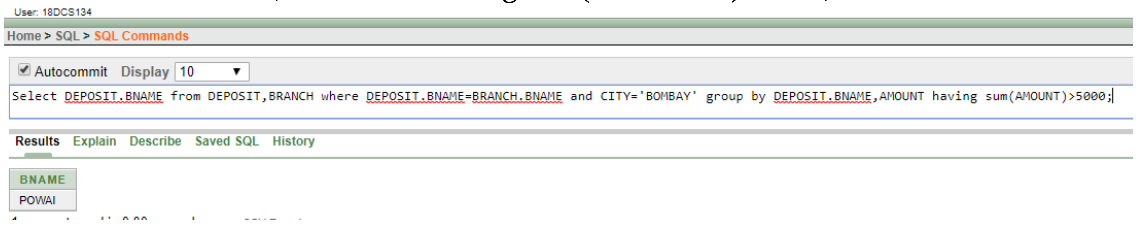

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
Select sum(EMP_SAL) from EMPLOYEE group by DEPT_NO having sum(EMP_SAL)>3000;
```

```
SUM(EMP_SAL)
11250
1 rows returned in 0.00 seconds
```

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

Query: Select DEPOSIT2.BNAME from DEPOSIT2,BRANCH1 where DEPOSIT2.BNAME=BRANCH1.BRANCH_NO and CITY='BOMBAY' group by DEPOSIT2.BNAME,AMOUNT having sum(AMOUNT)>5000;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
Select DEPOSIT.BNAME from DEPOSIT,BRANCH where DEPOSIT.BNAME=BRANCH.BNAME and CITY='BOMBAY' group by DEPOSIT.BNAME,AMOUNT having sum(AMOUNT)>5000;
```

```
BNAME
POWAI
```

Conclusion :

In this practical,we have studied different aggregation function like sum(), avg(), min(), max(), round() and also studied group by statement.

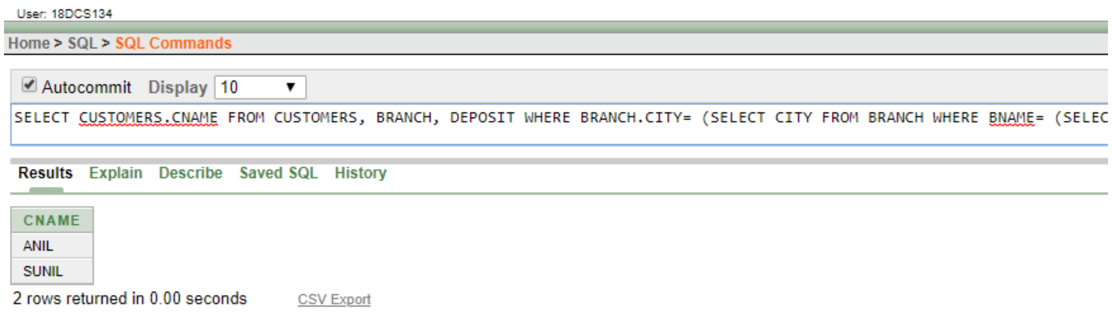
8. **Aim:-**

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

Query: SELECT EMP_NAME, HIRE_DATE FROM EMPLOYEE WHERE HIRE_DATE=(SELECT HIRE_DATE FROM EMPLOYEE WHERE EMP_NAME='Anita') AND EMP_NAME<>'Anita';

(2) Give name of customers who are depositors having same branch city of mr. sunil.

Query: SELECT CUSTOMERS.CNAME FROM CUSTOMERS, BRANCH, DEPOSIT WHERE BRANCH.CITY= (SELECT CITY FROM BRANCH WHERE BNAME= (SELECT BNAME FROM DEPOSIT WHERE CNAME='SUNIL')) AND DEPOSIT.BNAME=BRANCH.BNAME AND DEPOSIT.CNAME=CUSTOMERS.CNAME;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

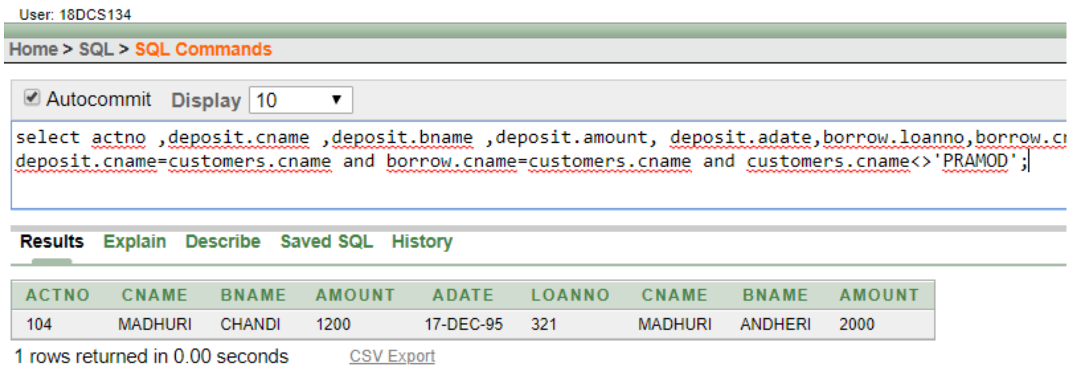
```
SELECT CUSTOMERS.CNAME FROM CUSTOMERS, BRANCH, DEPOSIT WHERE BRANCH.CITY= (SELECT CITY FROM BRANCH WHERE BNAME= (SELEC
```

CNAME
ANIL
SUNIL

2 rows returned in 0.00 seconds

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

Query: select actno ,deposit2.cname ,deposit2.bname ,deposit2.amount, deposit2.adate,borrow.loanno,borrow.cname,borrow.bname,borrow.amount from deposit2,customers,borrow where city=(select city from customers where cname='PRAMOD') and deposit2.cname=customers.cname and borrow.cname=customers.cname and customers.cname<>'PRAMOD';

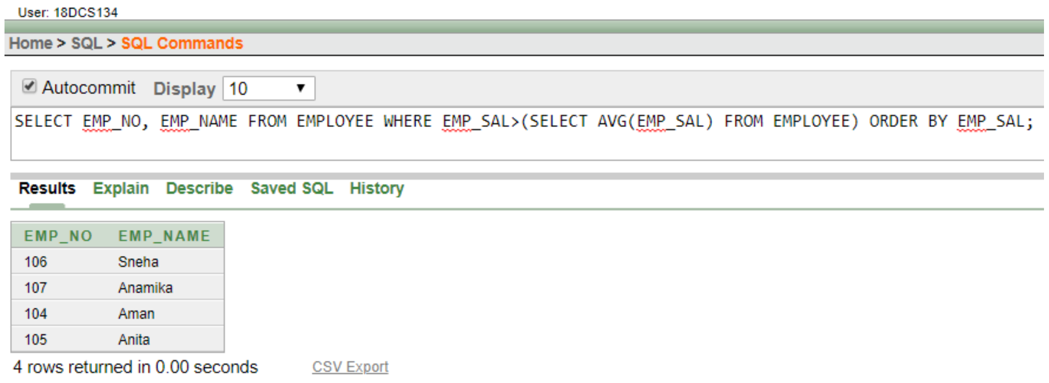

User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
select actno ,deposit.cname ,deposit.bname ,deposit.amount, deposit.adate,borrow.loanno,borrow.c
deposit.cname=customers.cname and borrow.cname=customers.cname and customers.cname<>'PRAMOD';
```

ACTNO	CNAME	BNAME	AMOUNT	ADATE	LOANNO	CNAME	BNAME	AMOUNT
104	MADHURI	CHANDI	1200	17-DEC-95	321	MADHURI	ANDHERI	2000

1 rows returned in 0.00 seconds

(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.
Query: SELECT EMP_NO, EMP_NAME FROM EMPLOYEE WHERE EMP_SAL>(SELECT AVG(EMP_SAL) FROM EMPLOYEE) ORDER BY EMP_SAL;

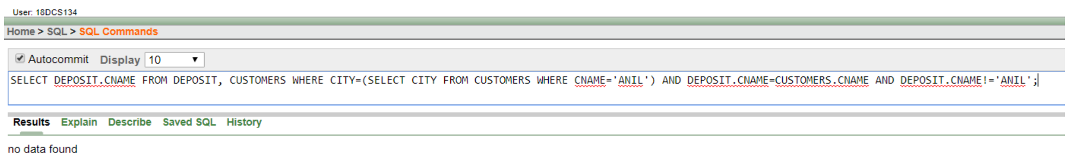


```
SELECT EMP_NO, EMP_NAME FROM EMPLOYEE WHERE EMP_SAL>(SELECT AVG(EMP_SAL) FROM EMPLOYEE) ORDER BY EMP_SAL;
```

EMP_NO	EMP_NAME
106	Sneha
107	Anamika
104	Aman
105	Anita

4 rows returned in 0.00 seconds

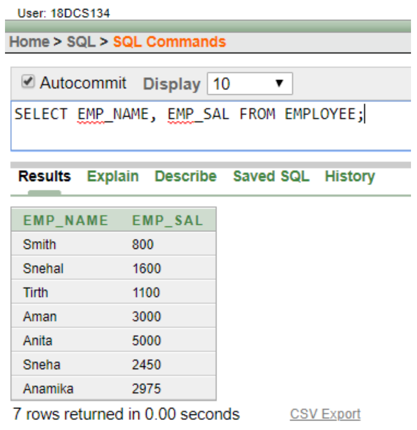
(5) Give names of depositors having same living city as mr. anil and having deposit amount greater than 2000
Query: SELECT DEPOSIT.CNAME FROM DEPOSIT, CUSTOMERS WHERE CITY=(SELECT CITY FROM CUSTOMERS WHERE CNAME='ANIL') AND DEPOSIT.CNAME=CUSTOMERS.CNAME AND DEPOSIT.CNAME!='ANIL';



```
SELECT DEPOSIT.CNAME FROM DEPOSIT, CUSTOMERS WHERE CITY=(SELECT CITY FROM CUSTOMERS WHERE CNAME='ANIL') AND DEPOSIT.CNAME=CUSTOMERS.CNAME AND DEPOSIT.CNAME!='ANIL';
```

no data found

(6) Display the last name and salary of every employee who reports to ford.
Query: SELECT EMP_NAME, EMP_SAL FROM EMPLOYEE;



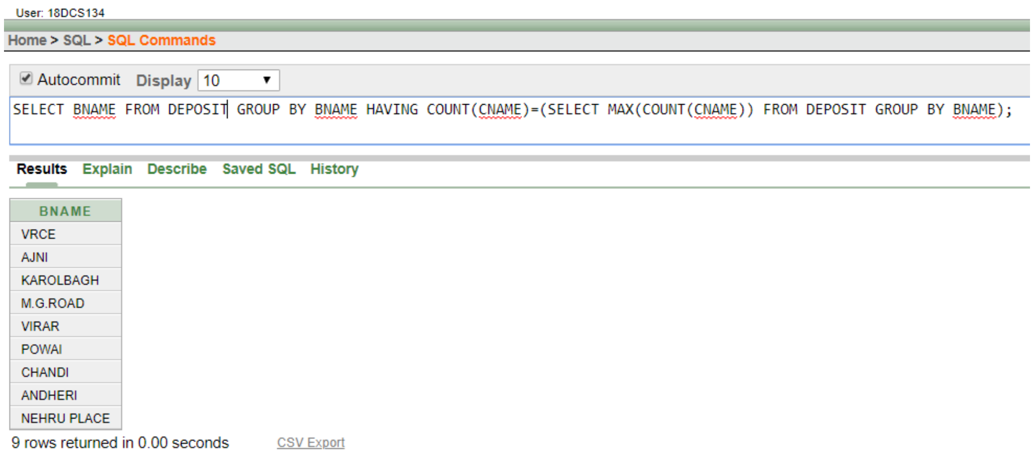
```
SELECT EMP_NAME, EMP_SAL FROM EMPLOYEE;
```

EMP_NAME	EMP_SAL
Smith	800
Snehal	1600
Tirth	1100
Aman	3000
Anita	5000
Sneha	2450
Anamika	2975

7 rows returned in 0.00 seconds

(7) Display the department number, name, and job for every employee in the Accounting department.
Query: SELECT JOB.DEPT_NO,DEPT_NAME,JOB_ID FROM EMPLOYEE, JOB WHERE JOB_TITLE='ACCOUNT' AND JOB.DEPT_NO=EMPLOYEE.DEPT_NO;

(8) List the name of branch having highest number of depositors.
Query: SELECT BNAME FROM DEPOSIT GROUP BY BNAME HAVING COUNT(CNAME)=(SELECT MAX(COUNT(CNAME)) FROM DEPOSIT GROUP BY BNAME);



User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

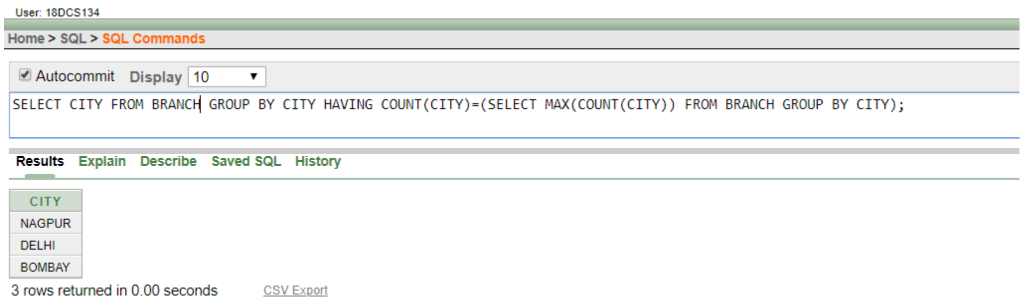
```
SELECT BNAME FROM DEPOSIT GROUP BY BNAME HAVING COUNT(CNAME)=(SELECT MAX(COUNT(CNAME)) FROM DEPOSIT GROUP BY BNAME);
```

Results Explain Describe Saved SQL History

BNAME
VRCE
AJNI
KAROLBAGH
M.G.ROAD
VIRAR
POWAI
CHANDI
ANDHERI
NEHRU PLACE

9 rows returned in 0.00 seconds CSV Export

(9) Give the name of cities where in which the maximum numbers of branches are located.
Query: SELECT CITY FROM BRANCH GROUP BY CITY HAVING COUNT(CITY)=(SELECT MAX(COUNT(CITY)) FROM BRANCH GROUP BY CITY);



User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10

```
SELECT CITY FROM BRANCH GROUP BY CITY HAVING COUNT(CITY)=(SELECT MAX(COUNT(CITY)) FROM BRANCH GROUP BY CITY);
```

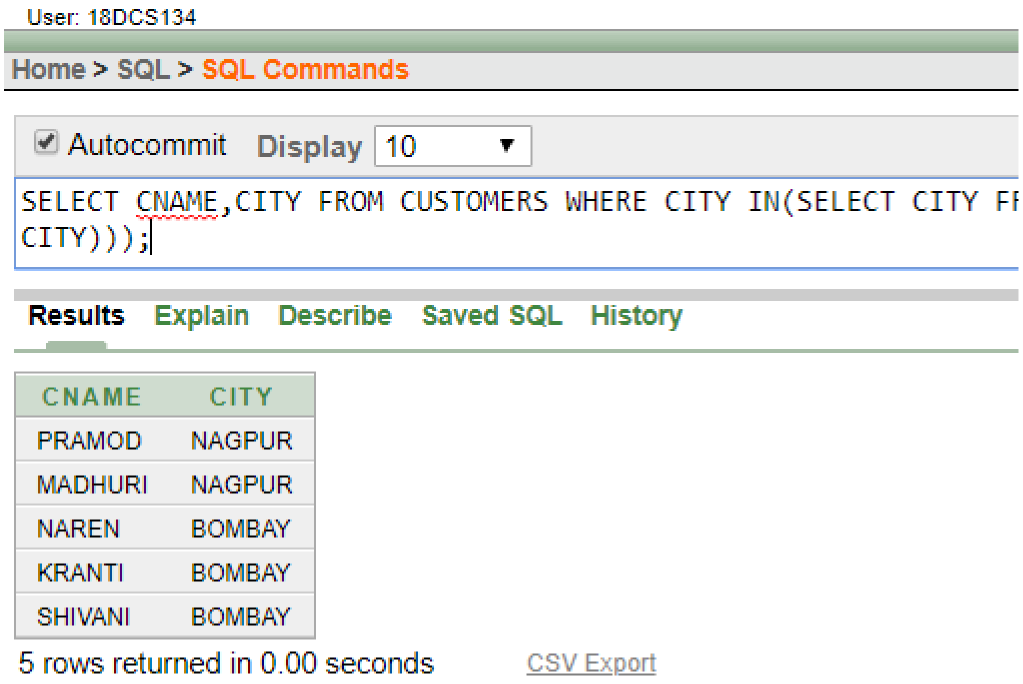
Results Explain Describe Saved SQL History

CITY
NAGPUR
DELHI
BOMBAY

3 rows returned in 0.00 seconds CSV Export

(10) Give name of customers living in same city where maximum depositors are located.
Query: SELECT CNAME,CITY FROM CUSTOMERS WHERE CITY IN(SELECT CITY FROM CUSTOMERS GROUP BY CITY HAVING COUNT(CITY)>1 AND CITY IN(SELECT CITY FROM BRANCH GROUP BY

CITY HAVING COUNT(CITY) = (SELECT MAX(COUNT(CITY)) FROM BRANCH GROUP BY CITY)));



```
SELECT CNAME,CITY FROM CUSTOMERS WHERE CITY IN(SELECT CITY FI
CITY)));
```

CNAME	CITY
PRAMOD	NAGPUR
MADHURI	NAGPUR
NAREN	BOMBAY
KRANTI	BOMBAY
SHIVANI	BOMBAY

5 rows returned in 0.00 seconds

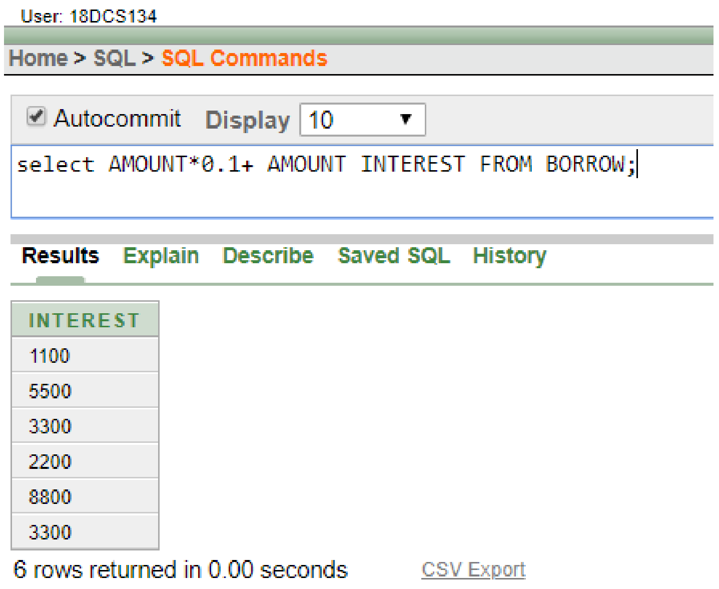
Conclusion:
In this practical, we have studied how to use subquery and its type-nested Sub query and Co-related subquery.

9. **Aim:-**

Manipulating Data:-

(1) Give 10% interest to all depositors.

Query: select AMOUNT*0.1+ AMOUNT INTEREST FROM BORROW;


User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
select AMOUNT*0.1+ AMOUNT INTEREST FROM BORROW;
```

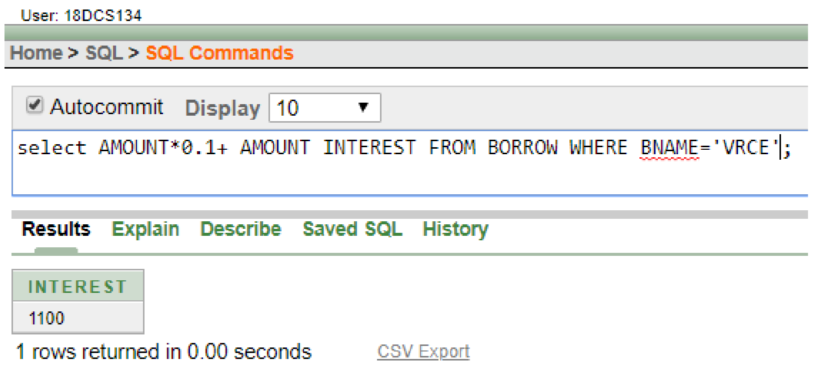
Results Explain Describe Saved SQL History

INTEREST
1100
5500
3300
2200
8800
3300

6 rows returned in 0.00 seconds CSV Export

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

Query: select AMOUNT*0.1+ AMOUNT INTEREST FROM BORROW WHERE BNAME='VRCE' ;


User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
select AMOUNT*0.1+ AMOUNT INTEREST FROM BORROW WHERE BNAME='VRCE';
```

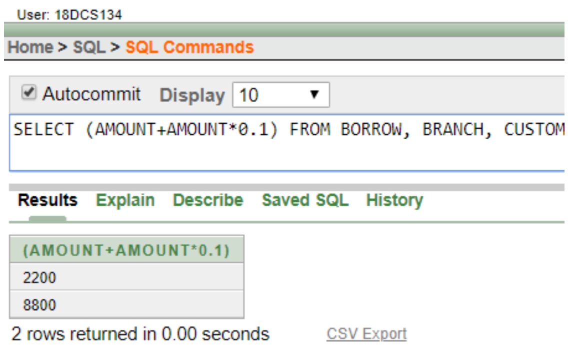
Results Explain Describe Saved SQL History

INTEREST
1100

1 rows returned in 0.00 seconds CSV Export

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

Query: 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';


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

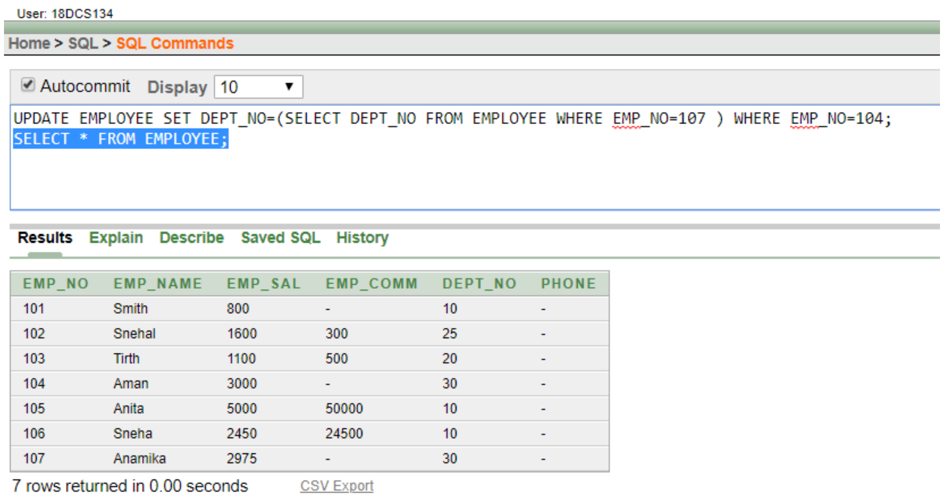
```
SELECT (AMOUNT+AMOUNT*0.1) FROM BORROW, BRANCH, CUSTOM
```

(AMOUNT+AMOUNT*0.1)
2200
8800

2 rows returned in 0.00 seconds

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

Query: UPDATE EMPLOYEE SET DEPT_NO=(SELECT DEPT_NO FROM EMPLOYEE WHERE EMP_NO=107) WHERE EMP_NO=104;
SELECT * FROM EMPLOYEE;


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

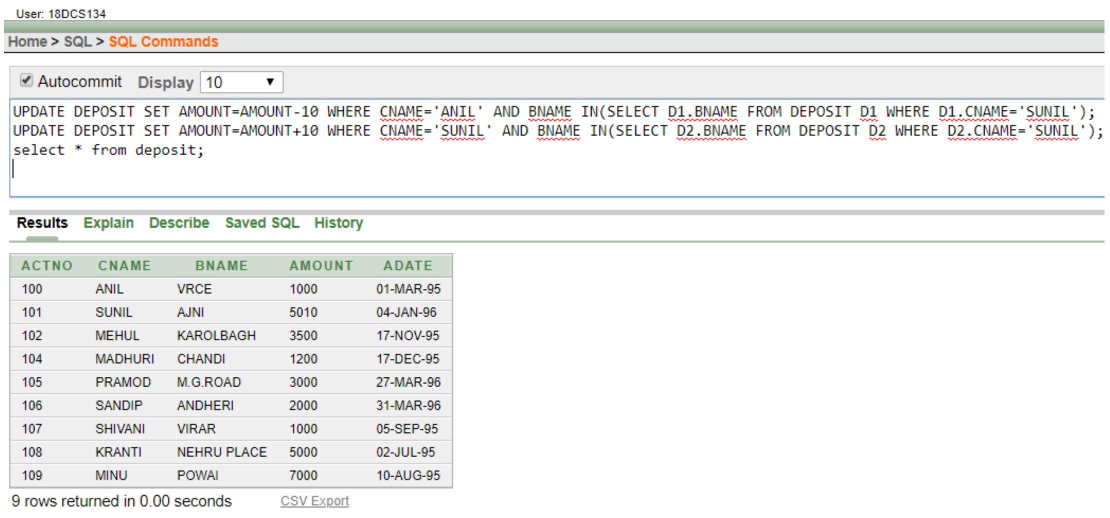
```
UPDATE EMPLOYEE SET DEPT_NO=(SELECT DEPT_NO FROM EMPLOYEE WHERE EMP_NO=107 ) WHERE EMP_NO=104;
SELECT * FROM EMPLOYEE;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	PHONE
101	Smith	800	-	10	-
102	Snehal	1600	300	25	-
103	Tirth	1100	500	20	-
104	Aman	3000	-	30	-
105	Anita	5000	50000	10	-
106	Sneha	2450	24500	10	-
107	Anamika	2975	-	30	-

7 rows returned in 0.00 seconds

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

Query: UPDATE DEPOSIT SET AMOUNT=AMOUNT-10 WHERE CNAME='ANIL' AND BNAME IN(SELECT D1.BNAME FROM DEPOSIT D1 WHERE D1.CNAME='SUNIL');
UPDATE DEPOSIT SET AMOUNT=AMOUNT+10 WHERE CNAME='SUNIL' AND BNAME IN(SELECT D2.BNAME FROM DEPOSIT D2 WHERE D2.CNAME='SUNIL');
select * from deposit;



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
UPDATE DEPOSIT SET AMOUNT=AMOUNT-10 WHERE CNAME='ANIL' AND BNAME IN(SELECT D1.BNAME FROM DEPOSIT D1 WHERE D1.CNAME='SUNIL');
UPDATE DEPOSIT SET AMOUNT=AMOUNT+10 WHERE CNAME='SUNIL' AND BNAME IN(SELECT D2.BNAME FROM DEPOSIT D2 WHERE D2.CNAME='SUNIL');
select * from deposit;
```

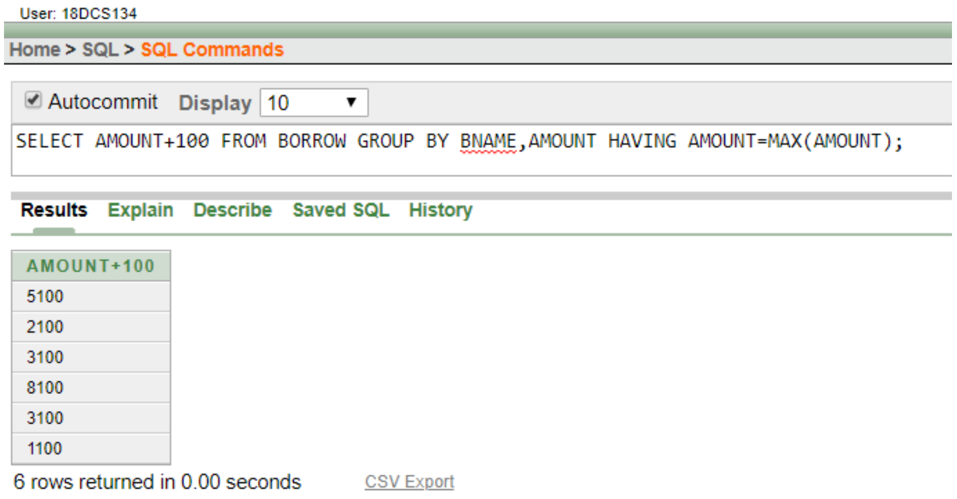
Results Explain Describe Saved SQL History

ACTNO	CNAME	BNAME	AMOUNT	ADATE
100	ANIL	VRCE	1000	01-MAR-95
101	SUNIL	AJNI	5010	04-JAN-96
102	MEHUL	KAROLBAGH	3500	17-NOV-95
104	MADHURI	CHANDI	1200	17-DEC-95
105	PRAMOD	M.G.ROAD	3000	27-MAR-96
106	SANDIP	ANDHERI	2000	31-MAR-96
107	SHIVANI	VIRAR	1000	05-SEP-95
108	KRANTI	NEHRU PLACE	5000	02-JUL-95
109	MINU	POWAI	7000	10-AUG-95

9 rows returned in 0.00 seconds CSV Export

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

Query: SELECT AMOUNT+100 FROM BORROW GROUP BY BNAME,AMOUNT HAVING AMOUNT=MAX(AMOUNT);



User: 18DCS134

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT AMOUNT+100 FROM BORROW GROUP BY BNAME,AMOUNT HAVING AMOUNT=MAX(AMOUNT);
```

Results Explain Describe Saved SQL History

AMOUNT+100
5100
2100
3100
8100
3100
1100

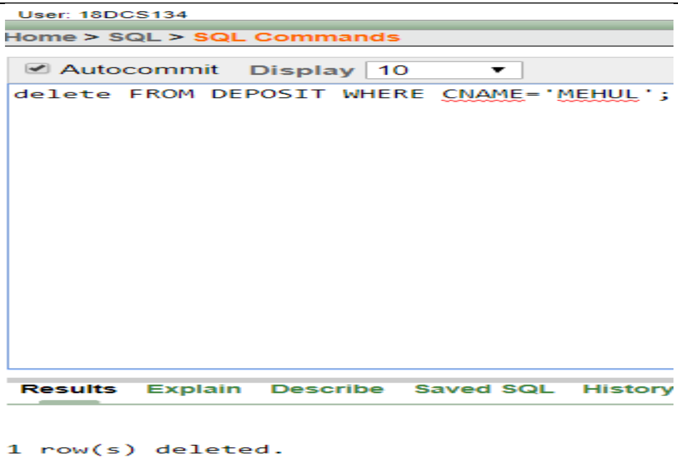
6 rows returned in 0.00 seconds CSV Export

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

Query: DELETE FROM DEPOSIT WHERE CNAME IN (SELECT D1.CNAME FROM DEPOSIT D1 GROUP BY D1.BNAME HAVING COUNT(D1.CNAME) BETWEEN 1 AND 3);

(8) Delete deposit of vijay.

Query: delete FROM DEPOSIT2 WHERE CNAME='MEHUL';



```
delete FROM DEPOSIT WHERE CNAME='MEHUL';
```

```
1 row(s) deleted.
```

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

Query: DELETE FROM BORROW WHERE CNAME IN(SELECT B.CNAME FROM BORROW B GROUP BY B.BNAME HAVING AVG(B.AMOUNT)<1000);

Conclusion: We have manipulated data using where condition statement.We have also studied how to use two or more table retrieve particular data.

10

Aim:-

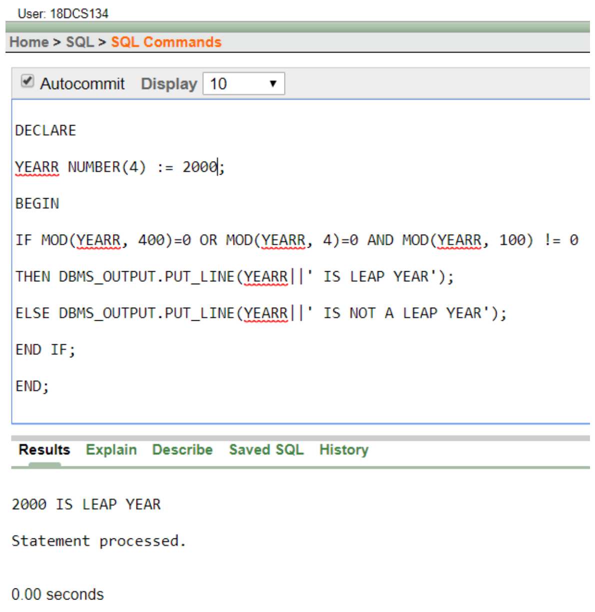
To perform basic PL/SQL blocks

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

Code:

```
DECLARE

YEARR NUMBER(4) := 2000;

BEGIN

IF MOD(YEARR, 400)=0 OR MOD(YEARR, 4)=0 AND MOD(YEARR, 100) != 0

THEN DBMS_OUTPUT.PUT_LINE(YEARR||' IS LEAP YEAR');

ELSE DBMS_OUTPUT.PUT_LINE(YEARR||' IS NOT A LEAP YEAR');

END IF;

END;
```

Output:-


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
DECLARE
YEARR NUMBER(4) := 2000;
BEGIN
IF MOD(YEARR, 400)=0 OR MOD(YEARR, 4)=0 AND MOD(YEARR, 100) != 0
THEN DBMS_OUTPUT.PUT_LINE(YEARR||' IS LEAP YEAR');
ELSE DBMS_OUTPUT.PUT_LINE(YEARR||' IS NOT A LEAP YEAR');
END IF;
END;
```

```
2000 IS LEAP YEAR

Statement processed.

0.00 seconds
```

Conclusion :

In this practical,we write PL/SQl block for checking leap year or not.

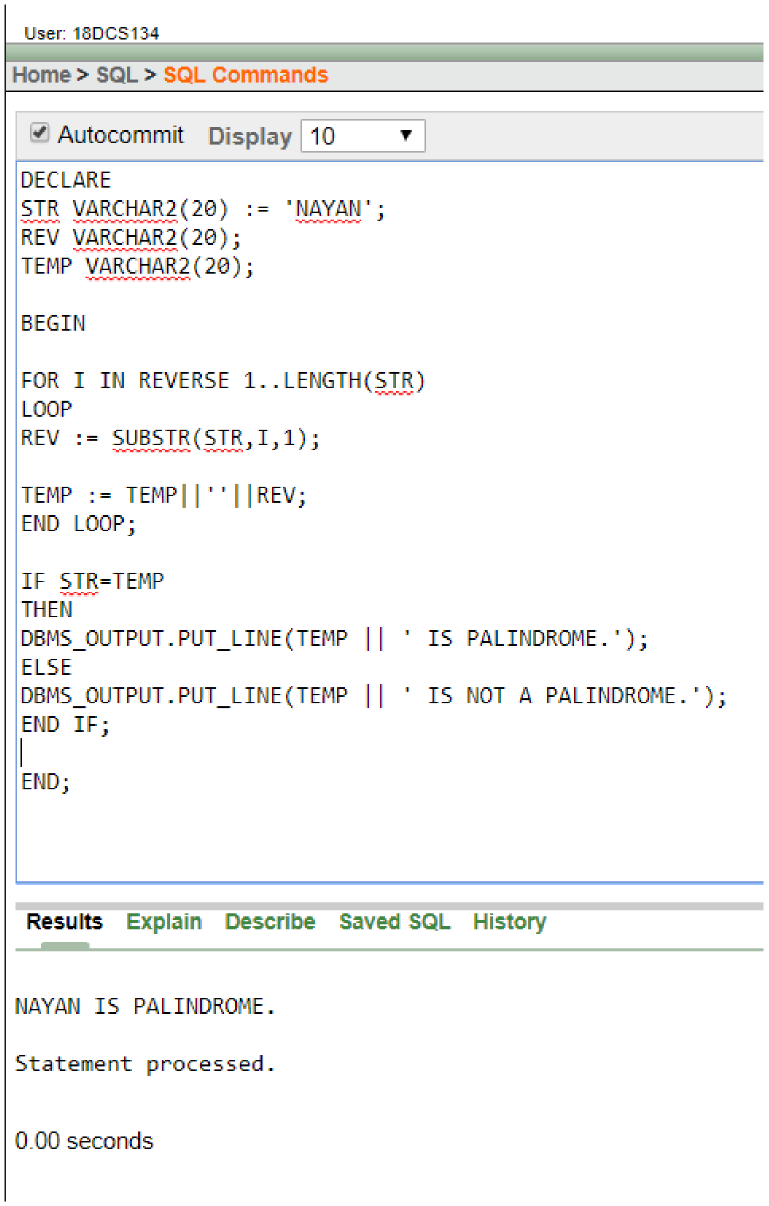
11. **Aim:-**

To perform the concept of loop
Find out whether given string is palindrome or not using for, While and Simple Loop.

Code:

```
DECLARE

STR VARCHAR2(20) := 'NAYAN';
REV VARCHAR2(20);
TEMP VARCHAR2(20);

BEGIN

FOR I IN REVERSE 1..LENGTH(STR)
LOOP
REV := SUBSTR(STR,I,1);

TEMP := TEMP||''||REV;
END LOOP;

IF STR=TEMP
THEN
DBMS_OUTPUT.PUT_LINE(TEMP ||' IS PALINDROME.');

ELSE DBMS_OUTPUT.PUT_LINE(TEMP ||' IS NOT A PALINDROME.');

END IF;

END;
```

Output:-


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
DECLARE
STR VARCHAR2(20) := 'NAYAN';
REV VARCHAR2(20);
TEMP VARCHAR2(20);

BEGIN

FOR I IN REVERSE 1..LENGTH(STR)
LOOP
REV := SUBSTR(STR,I,1);

TEMP := TEMP||''||REV;
END LOOP;

IF STR=TEMP
THEN
DBMS_OUTPUT.PUT_LINE(TEMP || ' IS PALINDROME.');
ELSE
DBMS_OUTPUT.PUT_LINE(TEMP || ' IS NOT A PALINDROME.');
END IF;

END;
```

```
NAYAN IS PALINDROME.

Statement processed.


0.00 seconds
```

Conclusion :
In this practical,we write PL/SQl block for checking string is palindrome or not.

12. **Aim:-**

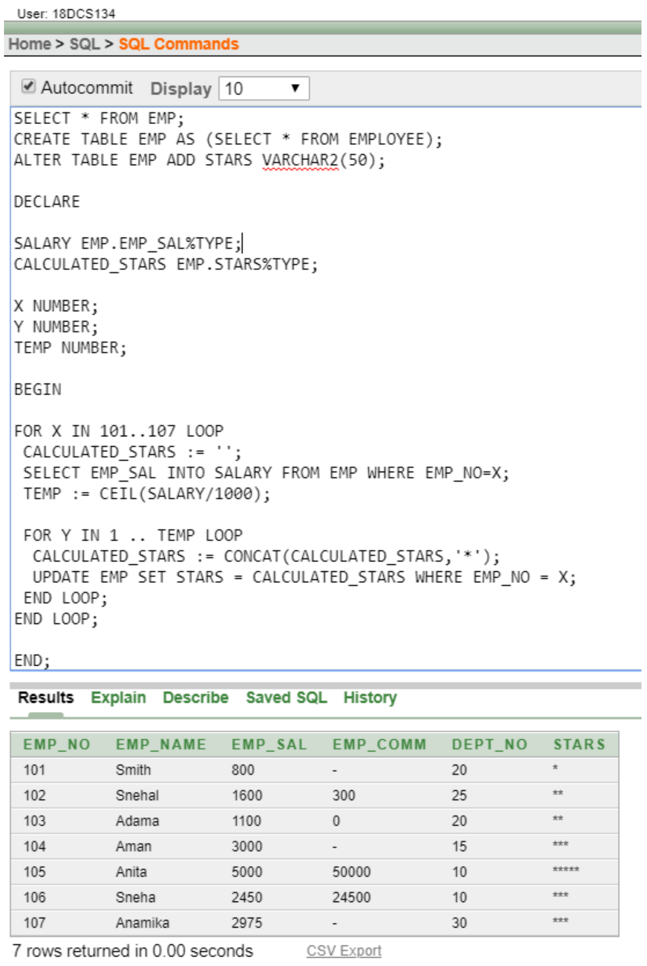
To understand the concept of “select into” and “% type” attribute.

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.

Code:

```
SELECT * FROM EMP;
CREATE TABLE EMP AS (SELECT * FROM EMPLOYEE);
ALTER TABLE EMP ADD STARS VARCHAR2(50);

DECLARE

SALARY EMP.EMP_SAL%TYPE;
CALCULATED_STARS EMP.STARS%TYPE;

X NUMBER;
Y NUMBER;
TEMP NUMBER;

BEGIN

FOR X IN 101..107 LOOP
 CALCULATED_STARS := '';
 SELECT EMP_SAL INTO SALARY FROM EMP WHERE EMP_NO=X;
 TEMP := CEIL(SALARY/1000);

 FOR Y IN 1 .. TEMP LOOP
  CALCULATED_STARS := CONCAT(CALCULATED_STARS,'*');
  UPDATE EMP SET STARS = CALCULATED_STARS WHERE EMP_NO = X;
 END LOOP;
END LOOP;

END;
```

Output:-


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM EMP;
CREATE TABLE EMP AS (SELECT * FROM EMPLOYEE);
ALTER TABLE EMP ADD STARS VARCHAR2(50);

DECLARE

SALARY EMP.EMP_SAL%TYPE;
CALCULATED_STARS EMP.STARS%TYPE;

X NUMBER;
Y NUMBER;
TEMP NUMBER;

BEGIN

FOR X IN 101..107 LOOP
 CALCULATED_STARS := '';
 SELECT EMP_SAL INTO SALARY FROM EMP WHERE EMP_NO=X;
 TEMP := CEIL(SALARY/1000);

 FOR Y IN 1 .. TEMP LOOP
  CALCULATED_STARS := CONCAT(CALCULATED_STARS,'*');
  UPDATE EMP SET STARS = CALCULATED_STARS WHERE EMP_NO = X;
 END LOOP;
END LOOP;

END;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO	STARS
101	Smith	800	-	20	*
102	Snehal	1600	300	25	**
103	Adama	1100	0	20	**
104	Aman	3000	-	15	***
105	Anita	5000	50000	10	*****
106	Sneha	2450	24500	10	***
107	Anamika	2975	-	30	***

7 rows returned in 0.00 seconds

Conclusion :

In this practical,we write PL/SQl block for adding star(1000 rs. salary per 1) in star column;

13.

Aim:

To perform the concept of cursor

Program Definition: (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.

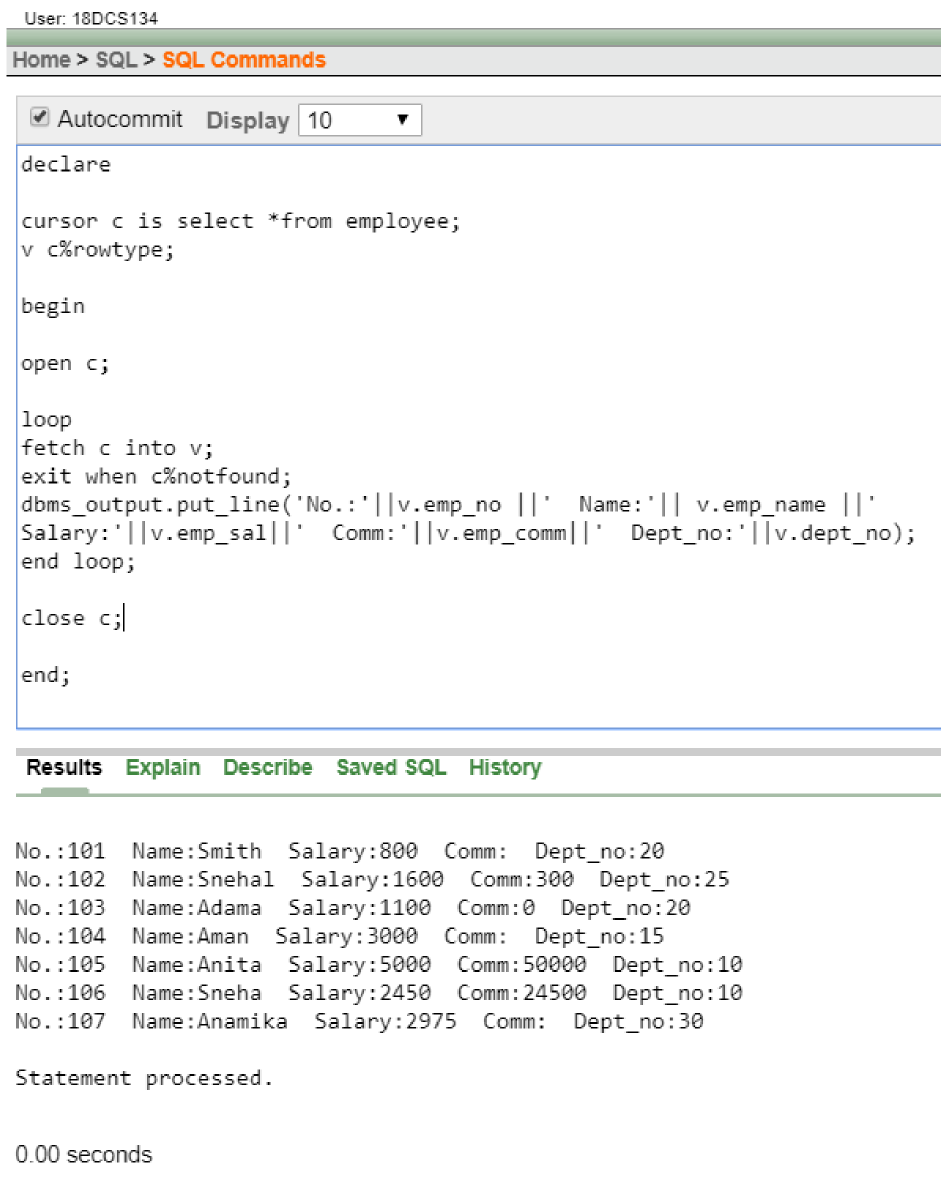
If the salary of the employee is less than 1000 and if the manager ID is either 7902 or 7839, display the message <<last name>> Due for a raise. Otherwise, display the message <<last_name>> Not due for a raise.

A))

Code:

```
declare

cursor c is select *from employee;
v c%rowtype;

begin

open c;

loop
fetch c into v;
exit when c%notfound;
dbms_output.put_line('No.:'||v.emp_no ||'  Name:'|| v.emp_name ||'  Salary:'||v.emp_sal||'
Comm:'||v.emp_comm||'  Dept_no:'||v.dept_no);
end loop;

close c;

end;
```

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
declare

cursor c is select *from employee;
v c%rowtype;

begin

open c;

loop
fetch c into v;
exit when c%notfound;
dbms_output.put_line('No.:'||v.emp_no ||'  Name:'|| v.emp_name ||'
Salary:'||v.emp_sal||'  Comm:'||v.emp_comm||'  Dept_no:'||v.dept_no);
end loop;

close c;

end;
```

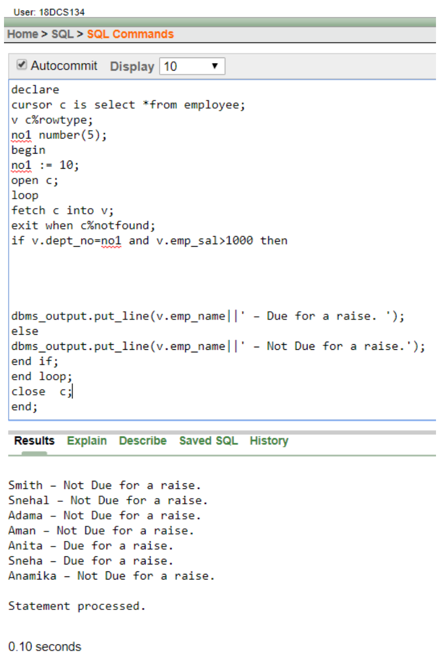
```
No.:101  Name:Smith  Salary:800  Comm:  Dept_no:20
No.:102  Name:Snehal  Salary:1600  Comm:300  Dept_no:25
No.:103  Name:Adama  Salary:1100  Comm:0  Dept_no:20
No.:104  Name:Aman  Salary:3000  Comm:  Dept_no:15
No.:105  Name:Anita  Salary:5000  Comm:50000  Dept_no:10
No.:106  Name:Sneha  Salary:2450  Comm:24500  Dept_no:10
No.:107  Name:Anamika  Salary:2975  Comm:  Dept_no:30

Statement processed.

0.00 seconds
```

Conclusion :
In this practical, we learned and perform about cursor in PL/SQL.

B)) Code:

```
declare
cursor c is select *from employee;
v c%rowtype;
no1 number(5);
begin
no1 := 10;
open c;
loop
fetch c into v;
exit when c%notfound;
if v.dept_no=no1 and v.emp_sal>1000 then

dbms_output.put_line(v.emp_name||' – Due for a raise. ');
else
dbms_output.put_line(v.emp_name||' – Not Due for a raise.');
end if;
end loop;
close  c;
end;
```

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
declare
cursor c is select *from employee;
v c%rowtype;
no1 number(5);
begin
no1 := 10;
open c;
loop
fetch c into v;
exit when c%notfound;
if v.dept_no=no1 and v.emp_sal>1000 then



dbms_output.put_line(v.emp_name||' - Due for a raise. ');
else
dbms_output.put_line(v.emp_name||' - Not Due for a raise.');
end if;
end loop;
close  c;
end;
```

```
Smith - Not Due for a raise.
Snehal - Not Due for a raise.
Adama - Not Due for a raise.
Aman - Not Due for a raise.
Anita - Due for a raise.
Sneha - Due for a raise.
Anamika - Not Due for a raise.

Statement processed.

0.10 seconds
```

14.

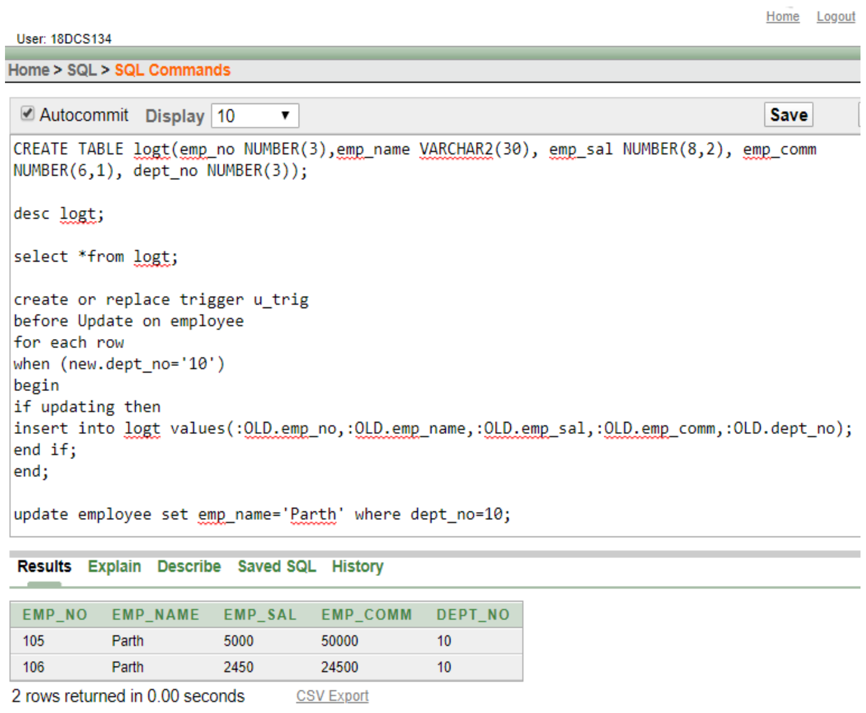
Aim: **To perform the concept of trigger**

Program Definition: Write a PL/SQL block to update the salary where deptno is 10. Generate trigger that will store the original record in other table before updation take place

Code:

```
CREATE TABLE logt(emp_no NUMBER(3),emp_name VARCHAR2(30), emp_sal
NUMBER(8,2), emp_comm NUMBER(6,1), dept_no NUMBER(3));

desc logt;

select *from logt;

create or replace trigger u_trig
before Update on employee
for each row
when (new.dept_no='10')
begin
if updating then
insert into logt
values(:OLD.emp_no,:OLD.emp_name,:OLD.emp_sal,:OLD.emp_comm,:OLD.dept_
no);
end if;
end;

update employee set emp_name='Parth' where dept_no=10;
```

Output:


Home Logout
User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Save
Results Explain Describe Saved SQL History
CSV Export

```
CREATE TABLE logt(emp_no NUMBER(3),emp_name VARCHAR2(30), emp_sal NUMBER(8,2), emp_comm
NUMBER(6,1), dept_no NUMBER(3));

desc logt;

select *from logt;

create or replace trigger u_trig
before Update on employee
for each row
when (new.dept_no='10')
begin
if updating then
insert into logt values(:OLD.emp_no,:OLD.emp_name,:OLD.emp_sal,:OLD.emp_comm,:OLD.dept_no);
end if;
end;

update employee set emp_name='Parth' where dept_no=10;
```

EMP_NO	EMP_NAME	EMP_SAL	EMP_COMM	DEPT_NO
105	Parth	5000	50000	10
106	Parth	2450	24500	10

2 rows returned in 0.00 seconds

Conclusion :

In this practical, we learned and perfomed trigger in PL/SQL.

15.

Aim: **To perform the concept of function and procedure.**

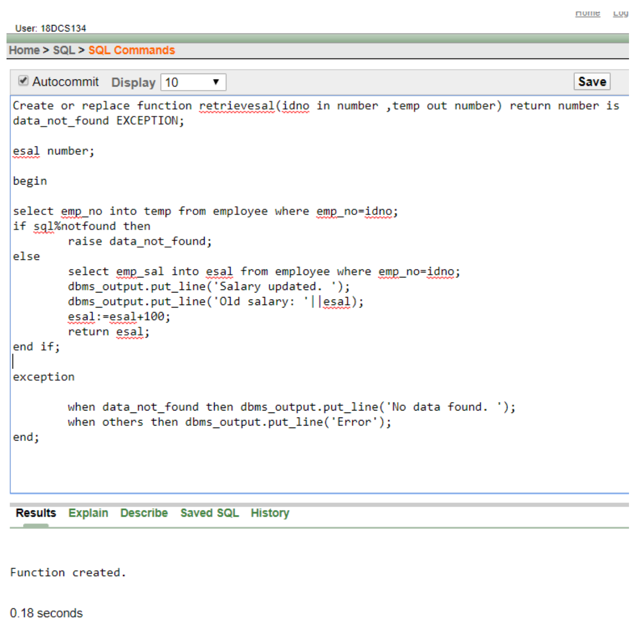
Program Definition: Write a PL/SQL block to update the salary of employee specified by empid. If record exist, then update the salary otherwise display appropriate message.
Write a function as well as procedure for updating salary.

Using Function:

Code:

```
Create or replace function retrievesal(idno in number ,temp out number) return number
is data_not_found EXCEPTION;

esal number;

begin

select emp_no into temp from employee where emp_no=idno;
if sql%notfound then
        raise data_not_found;
else
        select emp_sal into esal from employee where emp_no=idno;
        dbms_output.put_line('Salary updated. ');
        dbms_output.put_line('Old salary: '||esal);
        esal:=esal+100;
        return esal;
end if;

exception

        when data_not_found then dbms_output.put_line('No data found. ');
        when others then dbms_output.put_line('Error');
end;
```

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Save
Results Explain Describe Saved SQL History

```
Create or replace function retrievesal(idno in number ,temp out number) return number is
data_not_found EXCEPTION;

esal number;

begin

select emp_no into temp from employee where emp_no=idno;
if sql%notfound then
        raise data_not_found;
else
        select emp_sal into esal from employee where emp_no=idno;
        dbms_output.put_line('Salary updated. ');
        dbms_output.put_line('Old salary: '||esal);
        esal:=esal+100;
        return esal;
end if;

exception

        when data_not_found then dbms_output.put_line('No data found. ');
        when others then dbms_output.put_line('Error');
end;
```

```
Function created.

0.18 seconds
```

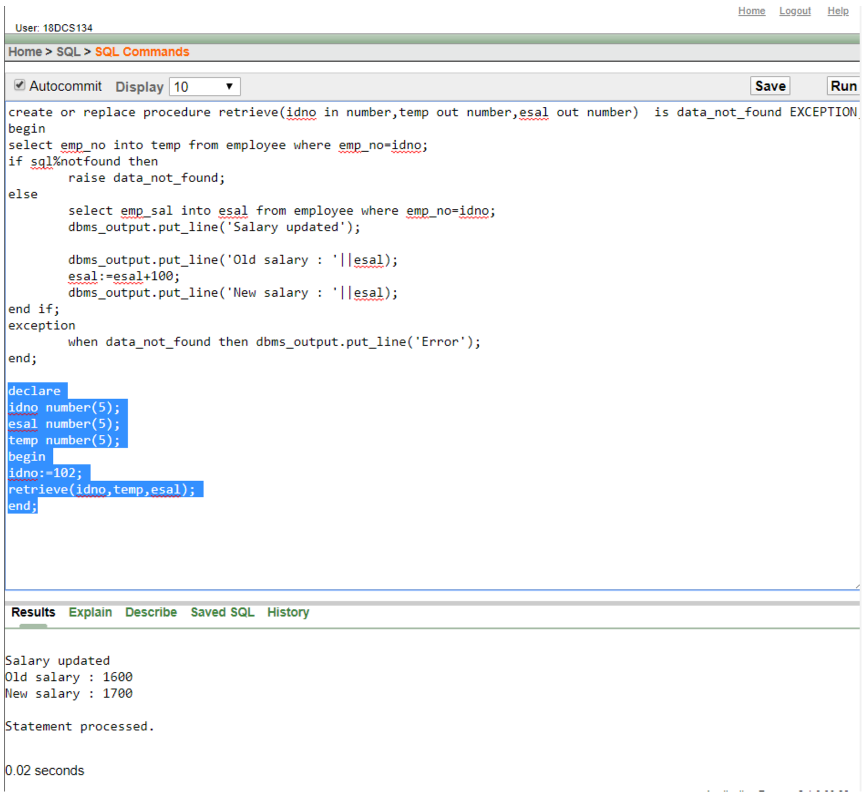
Using Procedure:

Code:

```
create or replace procedure retrieve(idno in number,temp out number,esal out number)
is data_not_found EXCEPTION;
begin
select emp_no into temp from employee where emp_no=idno;
if sql%notfound then
        raise data_not_found;
else
        select emp_sal into esal from employee where emp_no=idno;
        dbms_output.put_line('Salary updated');

        dbms_output.put_line('Old salary : '||esal);
        esal:=esal+100;
        dbms_output.put_line('New salary : '||esal);
end if;
exception
        when data_not_found then dbms_output.put_line('Error');
end;

declare
idno number(5);
esal number(5);
```

```
temp number(5);
begin
idno:=102;
retrieve(idno,temp,esal);
end;
```

Output:


Home Logout Help
User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Save Run
Results Explain Describe Saved SQL History

```
create or replace procedure retrieve(idno in number,temp out number,esal out number)  is data_not_found EXCEPTION
begin
select emp_no into temp from employee where emp_no=idno;
if sql%notfound then
        raise data_not_found;
else
        select emp_sal into esal from employee where emp_no=idno;
        dbms_output.put_line('Salary updated');

        dbms_output.put_line('Old salary : '||esal);
        esal:=esal+100;
        dbms_output.put_line('New salary : '||esal);
end if;
exception
        when data_not_found then dbms_output.put_line('Error');
end;

declare
idno number(5);
esal number(5);
temp number(5);
begin
idno:=102;
retrieve(idno,temp,esal);
end;
```

```
Salary updated
Old salary : 1600
New salary : 1700

Statement processed.


0.02 seconds
```

Conclusion :
In this practical, we learned and perform function and procedure in PL/SQL.

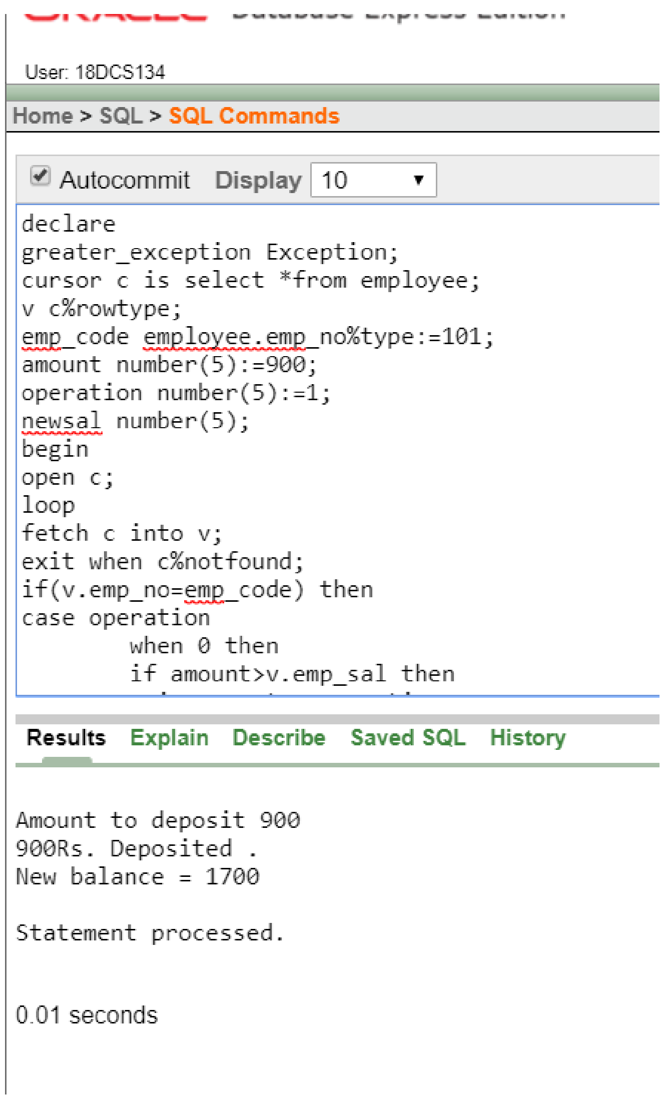
16.

Aim: To perform the concept of exception handler Write a PL/SQL block that will accept the employee code, amount and operation. Based on specified operation amount is added or deducted from salary of said employee. Use user defined exception handler for handling the exception.

Code:

```
declare
greater_exception Exception;
cursor c is select *from employee;
v c%rowtype;
emp_code employee.emp_no%type:=101;
amount number(5):=900;
operation number(5):=1;
newsal number(5);
begin
open c;
loop
fetch c into v;
exit when c%notfound;
if(v.emp_no=emp_code) then
case operation
        when 0 then
        if amount>v.emp_sal then
        raise greater_exception;
else
                newsal := v.emp_sal - amount;
                dbms_output.put_line('Amount to withdraw is '||amount);
                dbms_output.put_line(amount||'Rs.withdrawed. ');
                dbms_output.put_line('New balance = '||newsal);
        end if;
        when 1 then
                newsal := v.emp_sal + amount;
                dbms_output.put_line('Amount to deposit '||amount);

                dbms_output.put_line(amount||'Rs. Deposited . ');
                dbms_output.put_line('New balance = '||newsal);
        else
                dbms_output.put_line('Invalid expression.');
        end case;
end if;
end loop;
close c;
exception
when greater_exception then
dbms_output.put_line('Amount to withdraw = '||amount||' And balance = '||v.emp_sal);
dbms_output.put_line('Amount is greater . You can not withdraw');
when others then dbms_output.put_line('Error');
end;
```

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
declare
greater_exception Exception;
cursor c is select *from employee;
v c%rowtype;
emp_code employee.emp_no%type:=101;
amount number(5):=900;
operation number(5):=1;
newsal number(5);
begin
open c;
loop
fetch c into v;
exit when c%notfound;
if(v.emp_no=emp_code) then
case operation
        when 0 then
        if amount>v.emp_sal then
```

```
Amount to deposit 900
900Rs. Deposited .
New balance = 1700

Statement processed.


0.01 seconds
```

Conclusion :

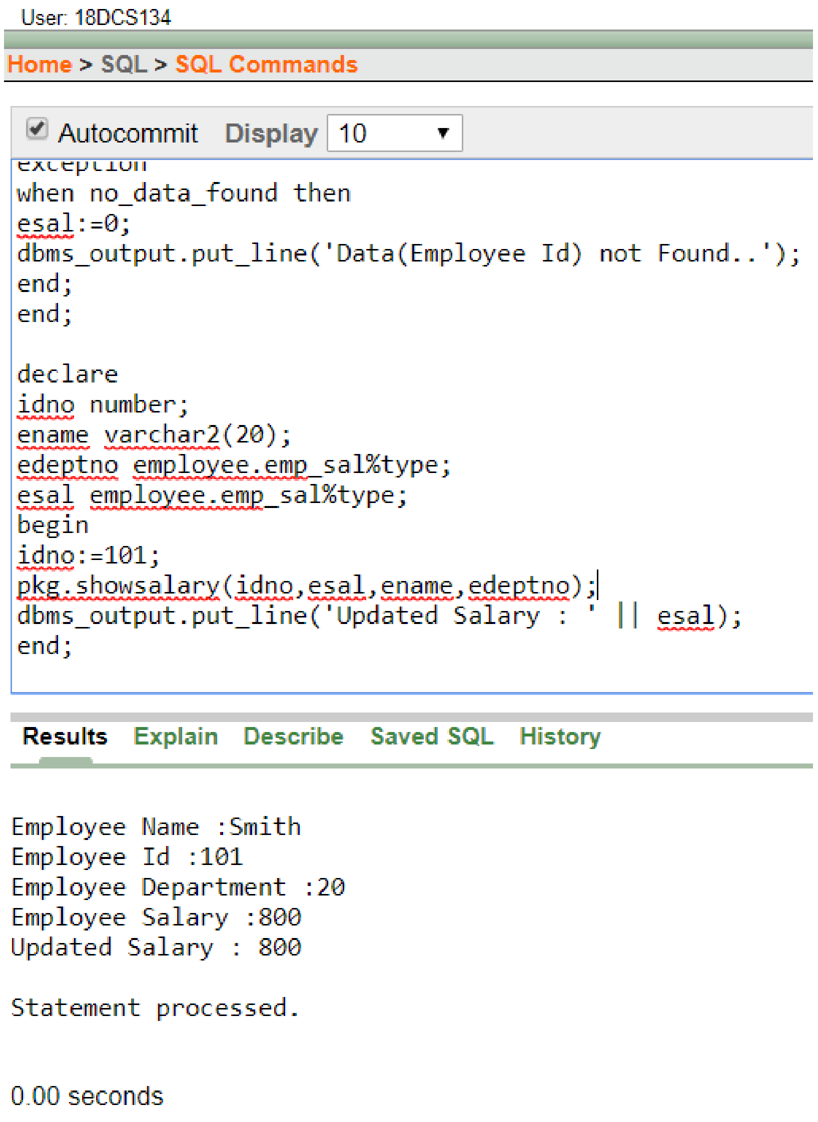
In this practical,we learned and perform about exception handler in PL/SQL.

17. **Aim: To perform the concept of package Create and invoke a package that contains private and public constructs.**

Code:

```
create or replace package pkg as
procedure showsalary(idno in number,esal out employee.emp_sal%type,ename out employee.emp_name%type,edeptno out number);
end;

create or replace package body pkg as
procedure showdepartment(idno in number,edeptno out employee.dept_no%type) is
begin
select dept_no into edeptno from employee where emp_no=idno;
exception
when no_data_found then
edeptno := 0;
end;
procedure showsalary(idno in number,esal out employee.emp_sal%type,ename out employee.emp_name%type,edeptno out number) is
begin
select emp_name into ename from employee where emp_no=idno;
showdepartment(idno,edeptno);
select emp_sal into esal from employee where emp_no=idno;
dbms_output.put_line('Employee Name :'||ename);
dbms_output.put_line('Employee Id :' ||idno);
dbms_output.put_line('Employee Department :'||edeptno);
dbms_output.put_line('Employee Salary :'||esal);
exception
when no_data_found then
esal:=0;
dbms_output.put_line('Data(Employee Id) not Found..');
end;
end;

declare
idno number;
ename varchar2(20);
edeptno employee.emp_sal%type;
esal employee.emp_sal%type;
begin
idno:=101;
pkg.showsalary(idno,esal,ename,edeptno);
dbms_output.put_line('Updated Salary : ' || esal);
end;
```

Output:


User: 18DCS134
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
exception
when no_data_found then
esal:=0;
dbms_output.put_line('Data(Employee Id) not Found..');
end;
end;

declare
idno number;
ename varchar2(20);
edeptno employee.emp_sal%type;
esal employee.emp_sal%type;
begin
idno:=101;
pkg.showsalary(idno,esal,ename,edeptno);
dbms_output.put_line('Updated Salary : ' || esal);
end;
```

```
Employee Name :Smith
Employee Id :101
Employee Department :20
Employee Salary :800
Updated Salary : 800

Statement processed.


0.00 seconds
```

Conclusion :
In this practical,we learned and performed concept of package in PL/SQL.