CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY
FACULTY OF TECHNOLOGY & ENGINEERING
DEVANG PATEL INSTITUE OF ADVANCE TECHNOLOGY AND RESEARCH

Subject: CE246: Database Management System
Semester: IV **ID:** 19DCE056 **Academic Year:** 2020-21

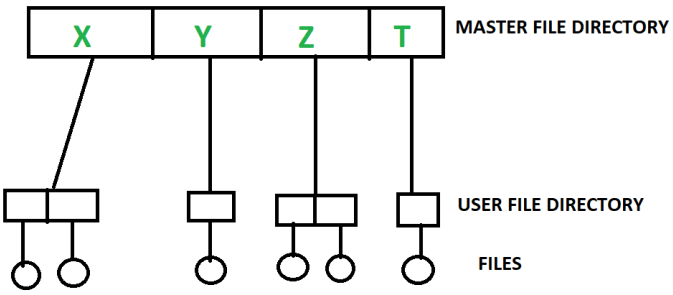
PRACTICAL - 1

AIM: Evaluation of Database (File System, DBMS, RDBMS, DDBMS)

THEORY:

1. File System: File system is basically a way of arranging the files in a storage medium like hard disk. File system organizes the files and helps in retrieval of files when they are required. File systems consists of different files which are grouped into directories. The directories further contain other folders and files. File system performs basic operations like management, file naming, giving access rules etc.
Example:

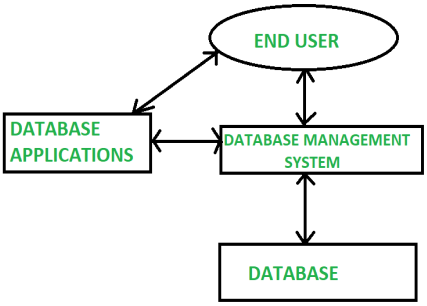
NTFS (New Technology File System), EXT (Extended File System).



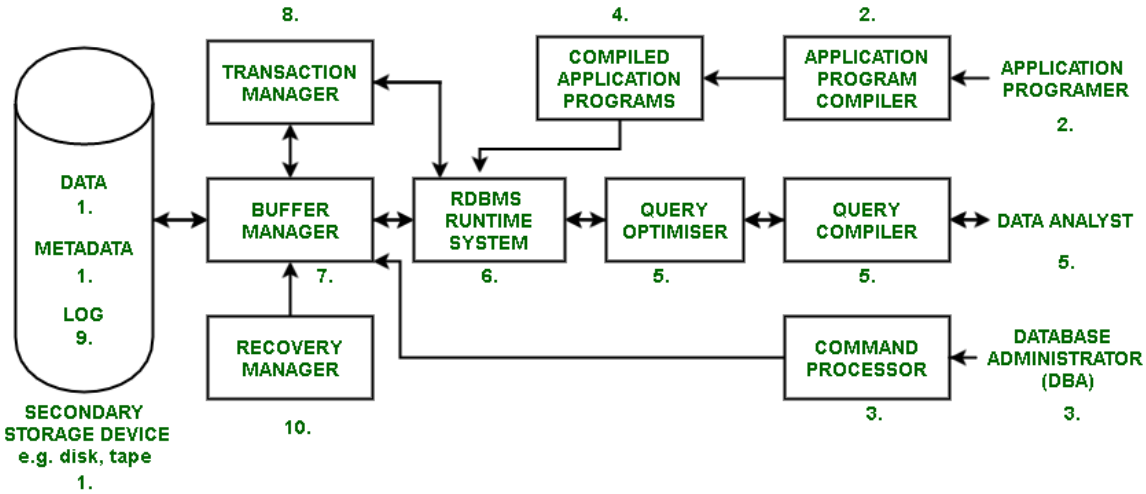
2. DBMS (Database Management System): Database Management System is basically a software that manages the collection of related data. It is used for storing data and retrieving the data effectively when it is needed. It also provides proper security measures for protecting the data from unauthorized access. In Database Management System the data can be fetched by SQL queries and relational algebra. It also provides mechanisms for data recovery and data backup.

Example:

Oracle, MySQL, MS SQL server.



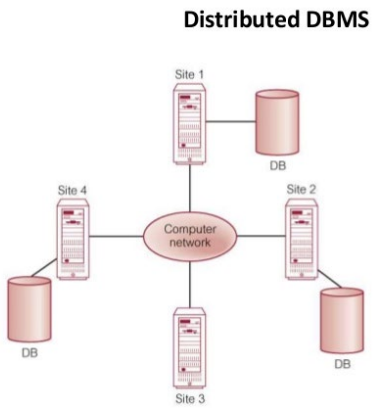
3. RDBMS: RDBMS stands for Relational Database Management System and it implements SQL. In the real-world scenario, people use the Relational Database Management System to collect information and process it, to provide service. E.g., In a ticket processing system, details about us (e.g., age, gender) and our journey (e.g. source, destination), are collected, and the ticket is provided to us.



RDBMS Architecture:

4. DDBMS:

A distributed database is basically a database that is not limited to one system, it is spread over different sites, i.e., on multiple computers or over a network of computers. A distributed database system is located on various sited that don't share physical components. This may



be required when a particular database needs to be accessed by various users globally. It needs to be managed such that for the users it looks like one single database.

CONCLUSION: From this practical we learned about File System, DBMS, RDBMS, DDBMS.

REFERENCE:

1. https://www.geeksforgeeks.org/difference-between-file-system-and-dbms/
2. https://www.geeksforgeeks.org/rdbms-architecture/
3. https://www.tutorialspoint.com/distributed_dbms/index.htm

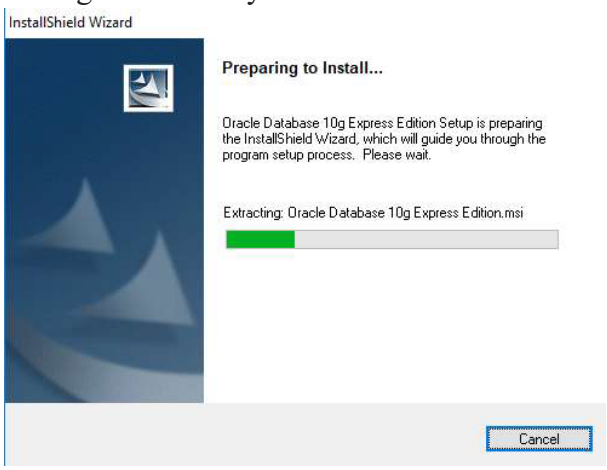
PRACTICAL - 2

AIM: Introduction to Oracle (step by step installation, introduction of sql, plsql).

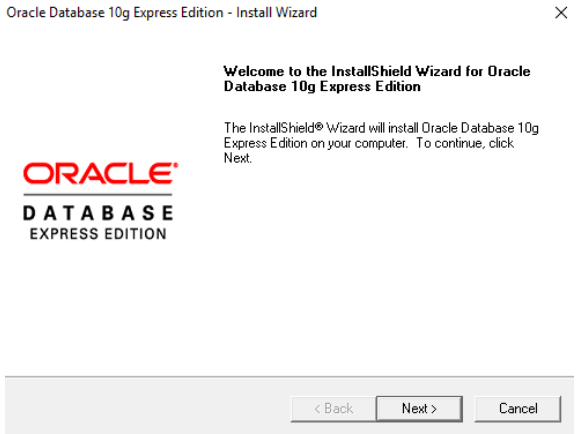
THEORY:

1) Download Oracle 10g from below link:
https://drive.google.com/file/d/1Y6ghDOEfVorTNrzWgF1UKaENHjeGgrHG/view

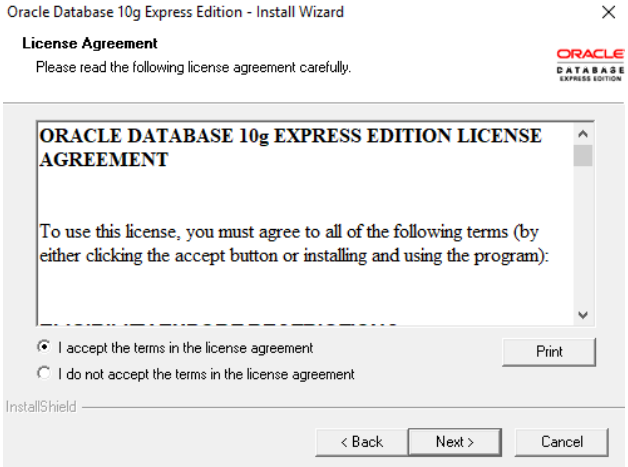
2) Install it by double clicking .exe which you have downloaded



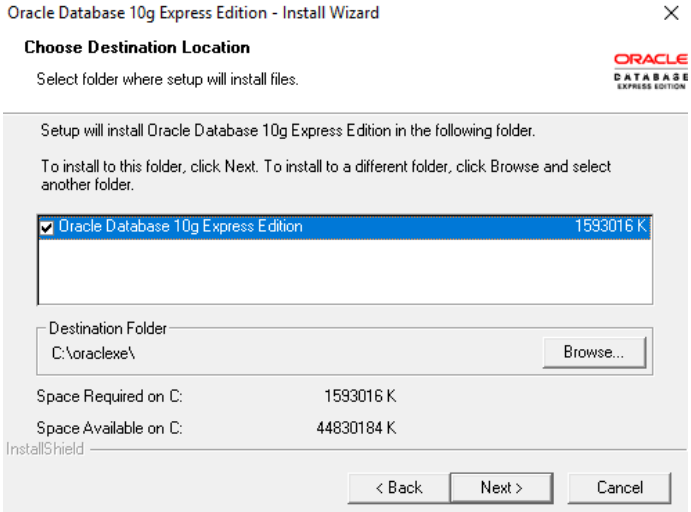
3) Click on Next button



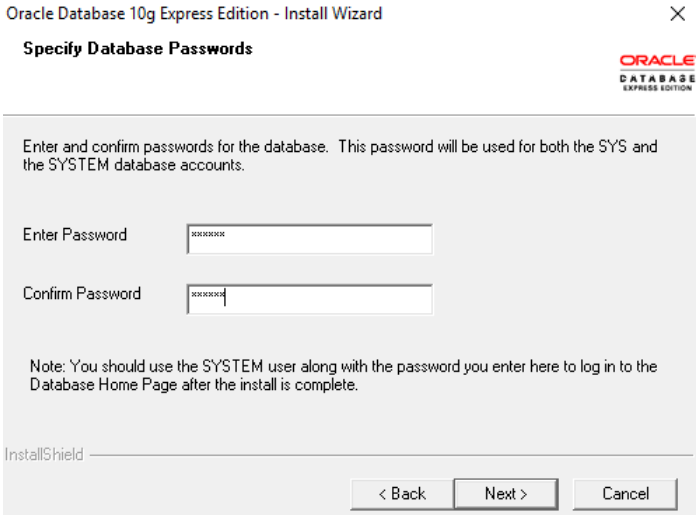
4) Accept license agreement and click on next button



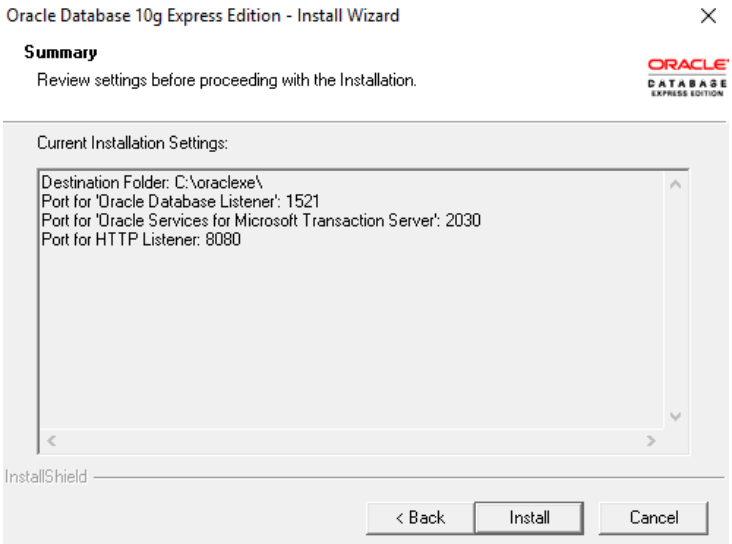
5) Click on next button



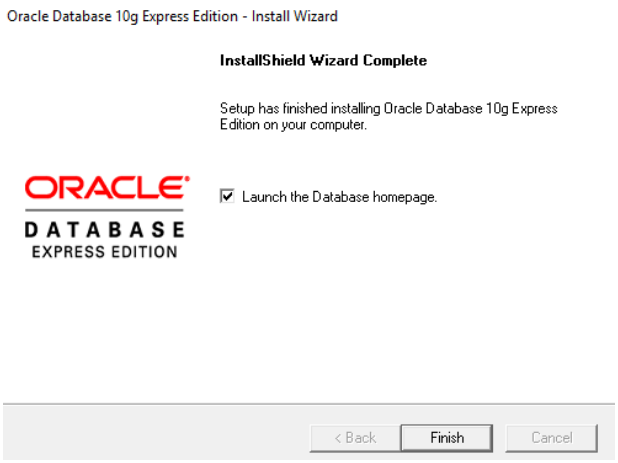
6) Enter password and confirm password for SYS and SYSTEM user. Please remember it because once installation will be over you have to enter it. To make it easy to remember give password as: “oracle”



7) Click on install button



8) Click on finish button.



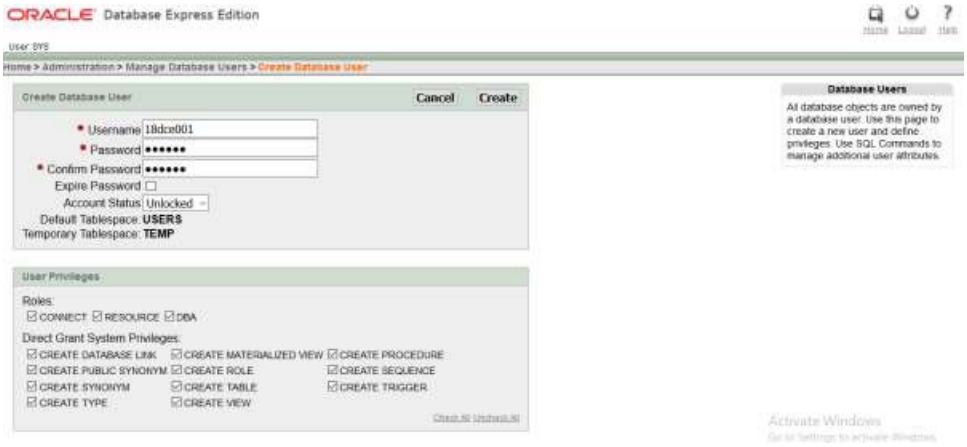
9) Enter username as SYS OR SYSTEM and enter your password (Entered in step: 6)

10) Click on Administration

11) Now click on “database user drop down button”. From that click on “create user”.

12) Enter your college roll no in username and give password (NEW) and confirm password. Don’t check expire password, make account status unblocked if it is not.

Give all privileges to your user. Finally click on “create” button.



13) This page will be shown to you. Now click on “logout” button.

14) Click on login

15) Enter username and password that you just created and click on “login” button
16) Click on SQL

17) Click on SQL Commands

18) Congratulation!!! Now you are ready to code SQL and PLSQL.

Introduction to SQL: Structured Query Language (SQL) is a standard Database language which is used to create, maintain and retrieve the relational database.

Introduction to PL/SQL: PL/SQL is a block structured language that enables developers to combine the power of SQL with procedural statements. All the statements of a block are passed to oracle engine all at once which increases processing speed and decreases the traffic. PL/SQL stands for "Procedural language extensions to SQL." PL/SQL is a database-oriented programming language that extends SQL with procedural capabilities. It was developed by Oracle Corporation within the early 90's to boost the capabilities of SQL.
PL/SQL adds selective (i.e., if…then…else…) and iterative constructs (i.e., loops) to SQL. PL/SQL is most helpful to put in writing triggers and keep procedures. Stored procedures square measure units of procedural code keep during a compiled type inside the info.

CONCLUSION: From this practical we installed Oracle 10 g learned about sql and pl/sql.

REFERENCE:

1. https://www.geeksforgeeks.org/structured-query-language/
2. https://www.geeksforgeeks.org/plsql-introduction/

PRACTICAL – 3

AIM: To study DDL-create and DML-insert commands.

COMMANDS:

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

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

Results Explain Describe Saved SQL History

```
Table created.
```

0.00 seconds

INSERT INTO DEPOSIT VALUES('100','ANIL','VRCE',1000.0,DATE '1995-03-01');
INSERT INTO DEPOSIT VALUES('101','SUNIL','AJNI',5000.0,DATE '1996-01-04');
INSERT INTO DEPOSIT VALUES('102','MEHUL','KAROLBAGH',3500.0,DATE '1995-11-17');
INSERT INTO DEPOSIT VALUES('104','MADHURI','CHANDI',1200.0,DATE '1995-12-17');
INSERT INTO DEPOSIT VALUES('105','PRAMOD','M.G.ROAD',3000.0,DATE '1996-03-27');
INSERT INTO DEPOSIT VALUES('106','SANDIP','ANDHERI',2000.0,DATE '1996-03-31');
INSERT INTO DEPOSIT VALUES('107','SHIVANI','VIRAR',1000.0,DATE '1995-09-05');
INSERT INTO DEPOSIT VALUES('108','KRANTI','NEHRU PLACE',5000.0,DATE '1995-07-02');
INSERT INTO DEPOSIT VALUES('109','MINU','POWAI',7000.0,DATE '1995-08-10');

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

INSERT INTO DEPOSIT VALUES('100','ANIL','VRCE',1000.0,DATE '1995-03-01');
INSERT INTO DEPOSIT VALUES('101','SUNIL','AJNI',5000.0,DATE '1996-01-04');
INSERT INTO DEPOSIT VALUES('102','MEHUL','KAROLBAGH',3500.0,DATE '1995-11-17');
INSERT INTO DEPOSIT VALUES('104','MADHURI','CHANDI',1200.0,DATE '1995-12-17');
INSERT INTO DEPOSIT VALUES('105','PRAMOD','M.G.ROAD',3000.0,DATE '1996-03-27');
INSERT INTO DEPOSIT VALUES('106','SANDIP','ANDHERI',2000.0,DATE '1996-03-31');
INSERT INTO DEPOSIT VALUES('107','SHIVANI','VIRAR',1000.0,DATE '1995-09-05');
INSERT INTO DEPOSIT VALUES('108','KRANTI','NEHRU PLACE',5000.0,DATE '1995-07-02');
INSERT INTO DEPOSIT VALUES('109','MINU','POWAI',7000.0,DATE '1995-08-10');
```

Results Explain Describe Saved SQL History

```
1 row(s) inserted.
```

0.00 seconds

```
INSERT INTO BRANCH VALUES('VRCE','NAGPUR');
INSERT INTO BRANCH VALUES('AJNI','NAGPUR');
INSERT INTO BRANCH VALUES('KAROLBAGH','DELHI');
INSERT INTO BRANCH VALUES('CHANDI','DELHI');
INSERT INTO BRANCH VALUES('DHARAMPETH','NAGPUR');
INSERT INTO BRANCH VALUES('M.G.ROAD','BANGLORE');
INSERT INTO BRANCH VALUES('ANDHERI','BOMBAY');
INSERT INTO BRANCH VALUES('VIRAR','BOMBAY');
INSERT INTO BRANCH VALUES('NEHRU PLACE','BOMBAY');
INSERT INTO BRANCH VALUES('POWAI','BOMBAY');
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

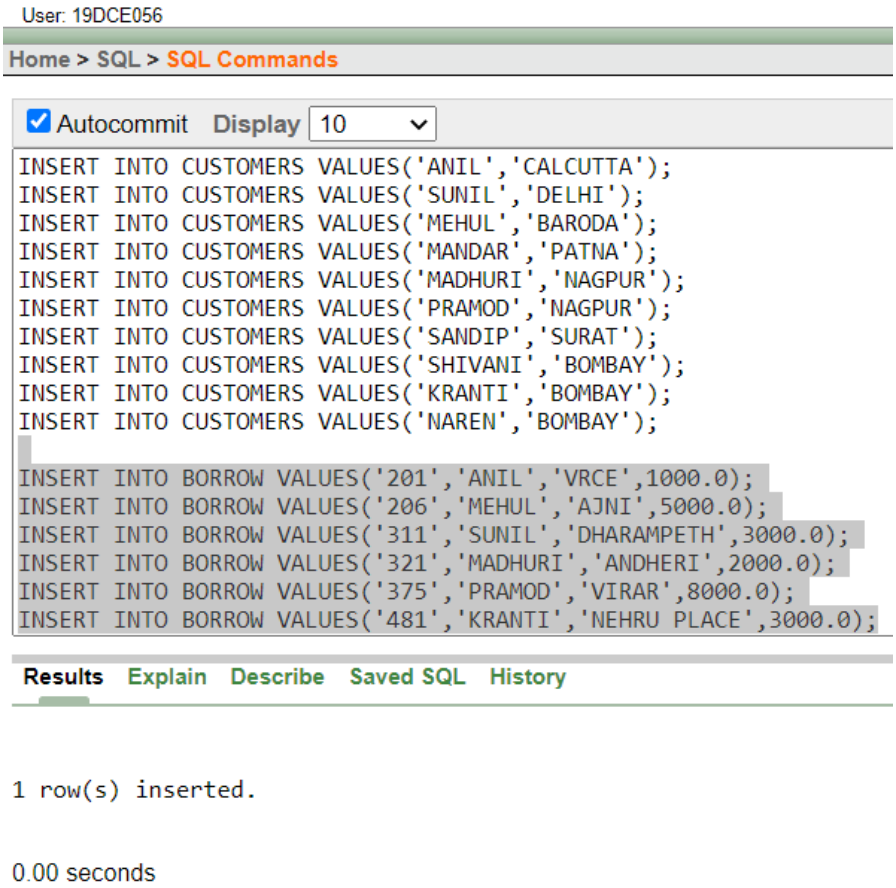
```
INSERT INTO BRANCH VALUES('M.G.ROAD','BANGLORE');
INSERT INTO BRANCH VALUES('ANDHERI','BOMBAY');
INSERT INTO BRANCH VALUES('VIRAR','BOMBAY');
INSERT INTO BRANCH VALUES('NEHRU PLACE','BOMBAY');
INSERT INTO BRANCH VALUES('POWAI','BOMBAY');

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

Results Explain Describe Saved SQL History

```
1 row(s) inserted.
```

0.00 seconds

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



INSERT INTO BORROW VALUES('201','ANIL','VRCE',1000.0);
INSERT INTO BORROW VALUES('206','MEHUL','AJNI',5000.0);
INSERT INTO BORROW VALUES('311','SUNIL','DHARAMPETH',3000.0);
INSERT INTO BORROW VALUES('321','MADHURI','ANDHERI',2000.0);
INSERT INTO BORROW VALUES('375','PRAMOD','VIRAR',8000.0);
INSERT INTO BORROW VALUES('481','KRANTI','NEHRU PLACE',3000.0);

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

INSERT INTO BORROW VALUES('201','ANIL','VRCE',1000.0);
INSERT INTO BORROW VALUES('206','MEHUL','AJNI',5000.0);
INSERT INTO BORROW VALUES('311','SUNIL','DHARAMPETH',3000.0);
INSERT INTO BORROW VALUES('321','MADHURI','ANDHERI',2000.0);
INSERT INTO BORROW VALUES('375','PRAMOD','VIRAR',8000.0);
INSERT INTO BORROW VALUES('481','KRANTI','NEHRU PLACE',3000.0);
```

Results Explain Describe Saved SQL History

```
1 row(s) inserted.
```

0.00 seconds

DESC DEPOSIT;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC DEPOSIT;
DESC BRANCH;
```

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

DESC BRANCH;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC DEPOSIT;
DESC BRANCH;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **BRANCH**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
BRANCH	BNAME	Varchar2	18	-	-	-	✓	-	-
	CITY	Varchar2	18	-	-	-	✓	-	-

1 - 2

DESC CUSTOMERS;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC CUSTOMERS;
DESC BORROW;
```

Results Explain Describe Saved SQL History

Object Type **TABLE** Object **CUSTOMERS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUSTOMERS	CNAME	Varchar2	18	-	-	-	✓	-	-
	CITY	Varchar2	18	-	-	-	✓	-	-

1 - 2

DESC BORROW;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC CUSTOMERS;
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

SELECT * FROM DEPOSIT;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC BORROW;

SELECT * FROM DEPOSIT;
SELECT * FROM BRANCH;
SELECT * FROM CUSTOMERS;
SELECT * FROM BORROW;
```

Results Explain Describe Saved SQL History

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

SELECT * FROM BRANCH;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC BORROW;

SELECT * FROM DEPOSIT;
SELECT * FROM BRANCH;
SELECT * FROM CUSTOMERS;
SELECT * FROM BORROW;
```

Results Explain Describe Save

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

10 rows returned in 0.01 seconds

SELECT * FROM CUSTOMERS;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC BORROW;

SELECT * FROM DEPOSIT;
SELECT * FROM BRANCH;
SELECT * FROM CUSTOMERS;
SELECT * FROM BORROW;
```

Results Explain Describe Sav

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 0.02 seconds

SELECT * FROM BORROW;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DESC BORROW;

SELECT * FROM DEPOSIT;
SELECT * FROM BRANCH;
SELECT * FROM CUSTOMERS;
SELECT * FROM BORROW;
```

Results Explain Describe Saved SQL History

LOANNO	CNAME	BNAME	AMOUNT
201	ANIL	VRCE	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 CSV Export

SELECT ACTNO,AMOUNT FROM DEPOSIT;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM DEPOSIT;
SELECT * FROM BRANCH;
SELECT * FROM CUSTOMERS;
SELECT * FROM BORROW;

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

SELECT * FROM DEPOSIT WHERE AMOUNT>4000;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM CUSTOMERS;
SELECT * FROM BORROW;

SELECT ACTNO,AMOUNT FROM DEPOSIT;

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

Results Explain Describe Saved SQL History

ACTNO	CNAME	BNAME	AMOUNT	ADATE
101	SUNIL	AJNI	5000	04-JAN-96
108	KRANTI	NEHRU PLACE	5000	02-JUL-95
109	MINU	POWAI	7000	10-AUG-95

3 rows returned in 0.00 seconds CSV Export

SELECT CNAME FROM DEPOSIT WHERE ADATE>DATE '1996-12-01';

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT ACTNO,AMOUNT FROM DEPOSIT;

SELECT * FROM DEPOSIT WHERE AMOUNT>4000;

SELECT CNAME FROM DEPOSIT WHERE ADATE>DATE '1996-12-01';
```

Results Explain Describe Saved SQL History

no data found

SELECT CITY FROM BRANCH WHERE BNAME='KAROLBAGH';

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT ACTNO,AMOUNT FROM DEPOSIT;

SELECT * FROM DEPOSIT WHERE AMOUNT>4000;

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

Results Explain Describe Saved SQL History

CITY
DELHI

1 rows returned in 0.00 seconds CSV Export

SELECT ACTNO,AMOUNT FROM DEPOSIT WHERE ADATE BETWEEN DATE '1996-12-01' AND DATE '1996-06-01';

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

SELECT CITY FROM BRANCH WHERE BNAME='KAROLBAGH';

SELECT ACTNO,AMOUNT FROM DEPOSIT WHERE ADATE BETWEEN DATE '1996-12-01' AND DATE '1996-06-01';
```

Results Explain Describe Saved SQL History

no data found

SELECT CNAME FROM DEPOSIT WHERE BNAME='VRCE';

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

SELECT ACTNO,AMOUNT FROM DEPOSIT WHERE ADATE E

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

Results Explain Describe Saved SQL History

CNAME
ANIL

1 rows returned in 0.00 seconds CSV Export

CONCLUSION: From this practical, we understood DDL commands and basic DML commands.

PRACTICAL – 4

AIM: **Create the below given table and insert the data accordingly**. **Perform following queries** and **study various options of LIKE predicate**

COMMANDS:

CREATE TABLE JOB(JOB_ID VARCHAR2(15), JOB_TITLE VARCHAR2(20), MIN_SAL NUMBER(7,2), MAX_SAL NUMBER(7,2));

CREATE TABLE EMPLOYEE(EMP_NO NUMBER(3), EMP_NAME VARCHAR2(30), EMP_SAL NUMBER(8,2), EMP_COM NUMBER(6,1), DEPT_NO NUMBER(3), L_NAME VARCHAR2(30), DEPT_NAME VARCHAR2(30), JOB_ID VARCHAR2(15),LOCATION VARCHAR2(15),MANAGER_ID NUMBER,(5)HIREDATE DATE);

INSERT INTO EMPLOYEE(EMP_NO, EMP_NAME, EMP_SAL, DEPT_NO, L_NAME, DEPT_NAME, JOB_ID, LOCATION, MANAGER_ID, HIREDATE) VALUES(101, 'SMITH', 800, 20, 'SHAH', 'MACHINE LEARNING', 'FI_MGR', 'TORONTO', 105, '9-AUG-96')

INSERT ALL
INTO EMPLOYEE(EMP_NO, EMP_NAME, EMP_SAL, EMP_COM, DEPT_NO, L_NAME, DEPT_NAME, JOB_ID, LOCATION, HIREDATE) VALUES(102, 'SNEHAL', 1600, 300, 25, 'GUPTA', 'DATA SCIENCE', 'LEC', 'LAS VEGAS', '14-MAR-96')
INTO EMPLOYEE VALUES(103, 'ADAMA', 1100, 0, 20, 'WALES', 'MACHINE LEARNING', 'MK_MGR', 'ONTARIO', 105, '30-NOV-95')
INTO EMPLOYEE(EMP_NO, EMP_NAME, EMP_SAL, DEPT_NO, L_NAME, DEPT_NAME, JOB_ID, LOCATION, MANAGER_ID, HIREDATE) VALUES(104, 'AMAN', 3000, 15, 'SHARMA', 'VIRUAL REALITY', 'COMP_OP', 'MEXICO', 12, '2-NOV-97')
INTO EMPLOYEE VALUES(105, 'ANITA', 5000, 50000, 10, 'PATEL', 'BIG DATA ANALYTICS', 'COMP_OP', 'GERMANY', 107, '01-JAN-98')
INTO EMPLOYEE VALUES(106, 'SNEHA', 2450, 24500, 10, 'JOSEPH', 'BIG DATA ANALYTICS', 'FI_ACC', 'MELBOURNE', 105, '26-sep-97')
INTO EMPLOYEE(EMP_NO, EMP_NAME, EMP_SAL, DEPT_NO, L_NAME, DEPT_NAME, JOB_ID, LOCATION, HIREDATE) VALUES(107, 'ANAMIKA', 2975, 30, 'JHA', 'ARTIFICIAL INTELLEGENCE', 'IT_PROG', 'NEW YORK', '15-JUL-97')
SELECT * FROM DUAL;

INSERT INTO JOB(JOB_ID, JOB_TITLE, MIN_SAL, MAX_SAL) VALUES('IT_PROG', 'Programmer', 4000, 10000)

INSERT ALL
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;

CREATE TABLE DEPOSIT1(A_NO VARCHAR2(5), CNAME VARCHAR2(15), BNAME VARCHAR2(10),AMOUNT NUMBER(7,2), A_DATE DATE)

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;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
INSERT ALL
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;

CREATE TABLE DEPOSIT1(A_NO  VARCHAR2(5), CNAME  VARCHAR2(15), BNAME VARCHAR2(10),AMOUNT NUMBER(7,2), A_DATE DATE)

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;
```

Results Explain Describe Saved SQL History

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

0.00 seconds

SELECT * FROM EMPLOYEE

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
INTO JOB VALUES('LEC', 'Lecturer', 6000, 17000)
INTO JOB VALUES('COMP_OP', 'Computer Operator', 1500, 3000)
SELECT * FROM DUAL;

CREATE TABLE DEPOSIT1(A_NO  VARCHAR2(5), CNAME  VARCHAR2(15), BNAME VARCHAR2(10),AMOUNT NUMBER(7,2), A_DATE DATE)

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;

SELECT * FROM EMPLOYEE
SELECT * FROM JOB
SELECT * FROM DEPOSIT1
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
101	SMITH	800	-	20	SHAH	MACHINE LEARNING	FI_MGR	TORONTO	105	09-AUG-96
102	SNEHAL	1600	300	25	GUPTA	DATA SCIENCE	LEC	LAS VEGAS	-	14-MAR-96
103	ADAMA	1100	0	20	WALES	MACHINE LEARNING	MK_MGR	ONTARIO	105	30-NOV-95
104	AMAN	3000	-	15	SHARMA	VIRUAL REALITY	COMP_OP	MEXICO	12	02-NOV-97
105	ANITA	5000	50000	10	PATEL	BIG DATA ANALYTICS	COMP_OP	GERMANY	107	01-JAN-98
106	SNEHA	2450	24500	10	JOSEPH	BIG DATA ANALYTICS	FI_ACC	MELBOURNE	105	26-SEP-97
107	ANAMIKA	2975	-	30	JHA	ARTIFICIAL INTELLEGENCE	IT_PROG	NEW YORK	-	15-JUL-97

SELECT * FROM JOB

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
INTO JOB VALUES('LEC', 'Lecturer', 6000, 17000)
INTO JOB VALUES('COMP_OP', 'Computer Operator',
SELECT * FROM DUAL;

CREATE TABLE DEPOSIT1(A_NO  VARCHAR2(5), CNAME

INSERT ALL
INTO DEPOSIT1 VALUES('101', 'ANIL', 'ANDHERI',
INTO DEPOSIT1 VALUES('102', 'SUNIL', 'VIRAR', 5
INTO DEPOSIT1 VALUES('103', 'JAY', 'VILLEPARLE'
INTO DEPOSIT1 VALUES('104', 'VIJAY', 'ANDHERI',
INTO DEPOSIT1 VALUES('105', 'KEYUR', 'DADAR', 7
INTO DEPOSIT1 VALUES('106', 'MAYUR', 'BORIVALI'
SELECT * FROM DUAL;

SELECT * FROM EMPLOYEE
SELECT * FROM JOB
SELECT * FROM DEPOSIT1
```

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
LEC	Lecturer	6000	17000
COMP_OP	Computer Operator	1500	3000

SELECT * FROM DEPOSIT1

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
INTO JOB VALUES('LEC', 'Lecturer', 6000, 17000)
INTO JOB VALUES('COMP_OP', 'Computer Operator',
SELECT * FROM DUAL;

CREATE TABLE DEPOSIT1(A_NO  VARCHAR2(5), CNAME

INSERT ALL
INTO DEPOSIT1 VALUES('101', 'ANIL', 'ANDHERI', 7
INTO DEPOSIT1 VALUES('102', 'SUNIL', 'VIRAR', 50
INTO DEPOSIT1 VALUES('103', 'JAY', 'VILLEPARLE',
INTO DEPOSIT1 VALUES('104', 'VIJAY', 'ANDHERI',
INTO DEPOSIT1 VALUES('105', 'KEYUR', 'DADAR', 75
INTO DEPOSIT1 VALUES('106', 'MAYUR', 'BORIVALI',
SELECT * FROM DUAL;

SELECT * FROM EMPLOYEE
SELECT * FROM JOB
SELECT * FROM DEPOSIT1
```

Results Explain Describe Saved SQL History

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

SELECT A_NO,AMOUNT FROM DEPOSIT1 WHERE A_DATE BETWEEN '01-JAN-06' AND '25-JUL-06'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM DUAL;

CREATE TABLE DEPOSIT1(A_NO  VARCHAR2(5), CNAME  VARCHAR2(15), BNAME VARCHAR2(10),A

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;

SELECT * FROM EMPLOYEE
SELECT * FROM JOB
SELECT * FROM DEPOSIT1

SELECT A_NO,AMOUNT FROM DEPOSIT1 WHERE A_DATE BETWEEN '01-JAN-06' AND '25-JUL-06'
```

Results Explain Describe Saved SQL History

A_NO	AMOUNT
101	7000
102	5000
103	6500

SELECT * FROM JOB WHERE MIN_SAL>4000

User: 19DCE056

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

SELECT EMP_NAME "EMPLOYEE",EMP_SAL "SALARY" FROM EMPLOYEE WHERE DEPT_NO=20

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT EMP_NAME "EMPLOYEE",EMP_SAL "SALARY" FROM EMPLOYEE WHERE DEPT_NO=20
```

Results Explain Describe Saved SQL History

EMPLOYEE	SALARY
SMITH	800
ADAMA	1100

SELECT EMP_NO, EMP_NAME, DEPT_NAME FROM EMPLOYEE WHERE DEPT_NO BETWEEN 10 AND 20

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT EMP_NAME "EMPLOYEE",EMP_SAL "SALARY" FROM EMPLOYEE WHERE DEPT_NO=20

SELECT EMP_NO, EMP_NAME, DEPT_NAME FROM EMPLOYEE WHERE DEPT_NO BETWEEN 10 AND 20
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	DEPT_NAME
101	SMITH	MACHINE LEARNING
103	ADAMA	MACHINE LEARNING
104	AMAN	VIRUAL REALITY
105	ANITA	BIG DATA ANALYTICS
106	SNEHA	BIG DATA ANALYTICS

SELECT CNAME||AMOUNT "DATA" FROM DEPOSIT1 WHERE AMOUNT!=8000

SELECT CONCAT(CNAME, AMOUNT) AS DATA FROM DEPOSIT1 WHERE AMOUNT!=8000

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT EMP_NO, EMP_NAME, DEPT_NAME FROM EMPLOYEE WHERE DEPT_NO BETWEEN

SELECT CONCAT(CNAME, AMOUNT) AS DATA FROM DEPOSIT1 WHERE AMOUNT!=8000
```

Results Explain Describe Saved SQL History

DATA
ANIL7000
SUNIL5000
JAY6500
KEYUR7500
MAYUR5500

SELECT * FROM EMPLOYEE WHERE EMP_COM IS NOT NULL AND MANAGER_ID IS NOT NULL

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM EMPLOYEE WHERE EMP_COM IS NOT NULL AND MANAGER_ID IS NOT NULL
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
103	ADAMA	1100	0	20	WALES	MACHINE LEARNING	MK_MGR	ONTARIO	105	30-NOV-95
105	ANITA	5000	50000	10	PATEL	BIG DATA ANALYTICS	COMP_OP	GERMANY	107	01-JAN-98
106	SNEHA	2450	24500	10	JOSEPH	BIG DATA ANALYTICS	FI_ACC	MELBOURNE	105	26-SEP-97

SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE 'A_a%'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM EMPLOYEE WHERE EMP_COM IS NOT NULL A

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

Results Explain Describe Saved SQL History

no data found

SELECT EMP_NAME, EMP_NO, EMP_SAL FROM EMPLOYEE WHERE EMP_NAME LIKE 'ANI__%'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

SELECT EMP_NAME, EMP_NO, EMP_SAL FROM EMPLOYEE WHERE EMP_NAME LIKE 'ANI__%'
```

Results Explain Describe Saved SQL History

EMP_NAME	EMP_NO	EMP_SAL
ANITA	105	5000

SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_N%' OR EMP_NAME LIKE '_M%'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT EMP_NAME, EMP_NO, EMP_SAL FROM EMPLOYEE WHERE EMP_NAME LIKE 'ANI__%'

SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_N%' OR EMP_NAME LIKE '_M%'
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
101	SMITH	800	-	20	SHAH	MACHINE LEARNING	FI_MGR	TORONTO	105	09-AUG-96
102	SNEHAL	1600	300	25	GUPTA	DATA SCIENCE	LEC	LAS VEGAS	-	14-MAR-96
104	AMAN	3000	-	15	SHARMA	VIRUAL REALITY	COMP_OP	MEXICO	12	02-NOV-97
105	ANITA	5000	50000	10	PATEL	BIG DATA ANALYTICS	COMP_OP	GERMANY	107	01-JAN-98
106	SNEHA	2450	24500	10	JOSEPH	BIG DATA ANALYTICS	FI_ACC	MELBOURNE	105	26-SEP-97
107	ANAMIKA	2975	-	30	JHA	ARTIFICIAL INTELLEGENCE	IT_PROG	NEW YORK	-	15-JUL-97

SELECT CNAME FROM DEPOSIT1 WHERE BNAME='ANDHERI' OR BNAME='DADAR' OR BNAME='VIRAR'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_N%' OR EMP_NAME LIKE '_M%'

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

Results Explain Describe Saved SQL History

CNAME
ANIL
SUNIL
VIJAY
KEYUR

SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE 'FI_%'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT CNAME FROM DEPOSIT1 WHERE BNAME='ANDHERI' OR

SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE 'FI_%'
```

Results Explain Describe Saved SQL History

JOB_TITLE
Finance Manager
Account

SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE '%_MGR' AND MAX_SAL>12000

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE 'FI_%'

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

Results Explain Describe Saved SQL History

JOB_TITLE
Marketing Manager

SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_N___'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_N___'
```

Results Explain Describe Saved SQL History

EMP_NO	EMP_NAME	EMP_SAL	EMP_COM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE
105	ANITA	5000	50000	10	PATEL	BIG DATA ANALYTICS	COMP_OP	GERMANY	107	01-JAN-98
106	SNEHA	2450	24500	10	JOSEPH	BIG DATA ANALYTICS	FI_ACC	MELBOURNE	105	26-SEP-97

SELECT EMP_COM,MANAGER_ID FROM EMPLOYEE WHERE EMP_COM IS NULL AND MANAGER_ID IS NULL AND EMP_NAME LIKE '__A%'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_N___'

SELECT EMP_COM,MANAGER_ID FROM EMPLOYEE WHERE EMP_COM IS NULL AND MANAGER_ID IS NULL AND EMP_NAME LIKE '__A%'
```

Results Explain Describe Saved SQL History

EMP_COM	MANAGER_ID
-	-

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

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT EMP_COM,MANAGER_ID FROM EMPLOYEE WHERE EMP_COM

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

CONCLUSION: After performing this practical, we understood the use of LIKE predicate and its variations.

PRACTICAL – 5

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

COMMANDS:

SELECT SUM(AMOUNT) FROM DEPOSIT

SQL Worksheet

```
1  SELECT SUM(AMOUNT) FROM DEPOSIT
2
3  SELECT SUM(AMOUNT) FROM BORROW W
4
```

SUM(AMOUNT)
28700

Download CSV

SELECT SUM(AMOUNT) FROM BORROW WHERE BNAME = 'karolbagh'

SQL Worksheet

```
1  SELECT SUM(AMOUNT) FROM DEPOSIT
2
3  SELECT SUM(AMOUNT) FROM BORROW WHERE BNAME = 'KAROLBAGH'
4
```

SUM(AMOUNT)
-

SELECT max (AMOUNT) FROM BORROW WHERE BNAME = 'vrce'

```
4
5  SELECT max (AMOUNT) FROM BORROW WHERE BNAME = 'VRCE'
6
```

MAX(AMOUNT)
1000

SELECT COUNT (CNAME) FROM CUSTOMERS

```
6
7  SELECT COUNT (CNAME) FROM CUSTOMERS
```

COUNT(CNAME)
10

SELECT COUNT (DISTINCT CITY) FROM CUSTOMERS

SQL Worksheet

```
SELECT COUNT (CNAME) FROM CUSTOMERS

SELECT COUNT (DISTINCT CITY) FROM CUSTOMERS
```

COUNT(DISTINCTCITY)
7

CREATE TABLE SUPPLIER AS SELECT * FROM EMPLOYEE

SQL Worksheet

```
SELECT COUNT (DISTINCT CITY) FROM CUSTOMERS

CREATE TABLE SUPPLIER AS SELECT * FROM EMPLOYEE
```

```
Table created.
```

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

SQL Worksheet

```
CREATE TABLE SUPPLIER AS SELECT * FROM EMPLOYEE

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

```
Table created.
```

CREATE TABLE SUPPLIER AS SELECT * FROM EMPLOYEE WHERE EMP_NAME ='NO NAME'

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

CREATE TABLE SUP2 AS SELECT * FROM EMPLOYEE WHERE EMP_NAME ='NO NAME'
```

```
Table created.
```

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

```
CREATE TABLE SUP2 AS SELECT * FROM EMPLOYEE WHERE EMP_NAME ='NO NAME

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

```
2 row(s) inserted.
```

DELETE SUP1

SQL Worksheet

```
INSERT INTO

DELETE SUP1
```

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

DELETE FROM SUP1 WHERE EMP_NO = 103

```
DELETE FROM SUP1 WHERE EMP_NO = 103
```

```
0 row(s) deleted.
```

RENAME SUP2 TO SUPP0
ALTER TABLE SUP2 RENAME TO SUPP0

```
DELETE FROM SUP1 WHERE

RENAME SUP2 TO SUPP0
```

```
Statement processed.
```

DROP TABLE SUP1

```
DROP TABLE SUP1
```

```
Table dropped.
```

UPDATE EMPLOYEE SET DEPT_NO=10 WHERE EMP_NAME LIKE '_M%'

```
DROP TABLE SUP1

update employee set DEPT_NO=10 where EMP_NAME like '_M%'
```

```
2 row(s) updated.
```

UPDATE EMPLOYEE SET EMP_NAME='UPDATED NAME' WHERE EMP_NO = 103

```
UPDATE EMPLOYEE SET EMP_NAME='UPDATED NAME' WHERE EMP_NO = 103
```

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

ALTER TABLE EMPLOYEE ADD PHONE NUMBER(10)

```
33  alter table EMPLOYEE add phone number(10)
```

```
Table altered.
```

ALTER TABLE EMPLOYEE MODIFY EMP_NAME VARCHAR2(30)

```
35  alter table employee modify emp_name varchar2(30)
```

```
Table altered.
```

SELECT COUNT(DISTINCT DEPT_NO) FROM EMPLOYEE WHERE EMP_SAL > 1000

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

COUNT(DISTINCTDEPT_NO)
4

SELECT * FROM EMPLOYEE ORDER BY EMP_NAME ASC , EMP_NO DESC

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

EMP_NO	EMP_NAME	EMP_SAL	EMP_COM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE	PHONE
104	AMAN	3000	-	10	SHARMA	VIRUAL REALITY	COMP_OP	MEXICO	12	02-NOV-97	-
107	ANAMIKA	2975	-	30	JHA	ARTIFICIAL INTELLEGENCE	IT_PROG	NEW YORK	-	15-JUL-97	-
105	ANITA	5000	50000	10	PATEL	BIG DATA ANALYTICS	COMP_OP	GERMANY	107	01-JAN-98	-
101	SMITH	800	-	10	SHAH	MACHINE LEARNING	FI_MGR	TORONTO	105	09-AUG-96	-
106	SNEHA	2450	24500	10	JOSEPH	BIG DATA ANALYTICS	FI_ACC	MELBOURNE	105	26-SEP-97	-
102	SNEHAL	1600	300	25	GUPTA	DATA SCIENCE	LEC	LAS VEGAS	-	14-MAR-96	-
103	UPDATED NAME	1100	0	20	WALES	MACHINE LEARNING	MK_MGR	ONTARIO	105	30-NOV-95	-

SELECT DISTINCT DEPT_NO FROM EMPLOYEE ORDER BY DEPT_NO ASC

```
41  select distinct dept_no from employee order by dept_no asc
```

DEPT_NO
10
20
25
30

SELECT EMP_COMM FROM EMPLOYEE ORDER BY EMP_COMM DESC NULLS LAST

```
43  select emp_com from employee order by emp_com desc nulls last
44
45  update employee set emp_comm = 500 where dept_no = 20
46
```

EMP_COM
50000
24500
300
0
-
-
-

UPDATE EMPLOYEE SET EMP_COMM = 500 WHERE DEPT_NO = 20

```
45  update employee set emp_com = 500 where dept_no = 20
46
```

1 row(s) updated.

SELECT EMP_COMM FROM EMPLOYEE ORDER BY EMP_COMM ASC NULLS FIRST

```
47  select emp_com from employee order by emp_com asc nulls first
48  select emp_sal from employee order by emp_sal desc nulls first
49
```

EMP_COM
-
-
-
300
500
24500
50000

SELECT EMP_SAL FROM EMPLOYEE ORDER BY EMP_SAL DESC NULLS FIRST

```
48 select emp_sal from employee order by emp_sal desc nulls first
49
```

EMP_SAL
5000
3000
2975
2450
1600
1100
800

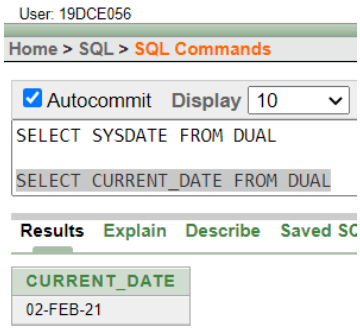
CONCLUSION: After performing this practical, we understood various DML commands, aggregate functions and sorting concept on all created tables.

PRACTICAL – 6

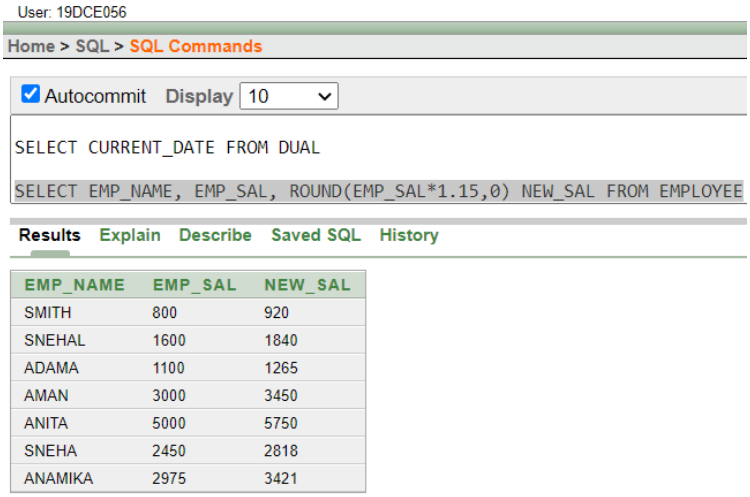
AIM: To study Single-row functions.

COMMANDS:

SELECT SYSDATE FROM DUAL
SELECT CURRENT_DATE FROM DUAL

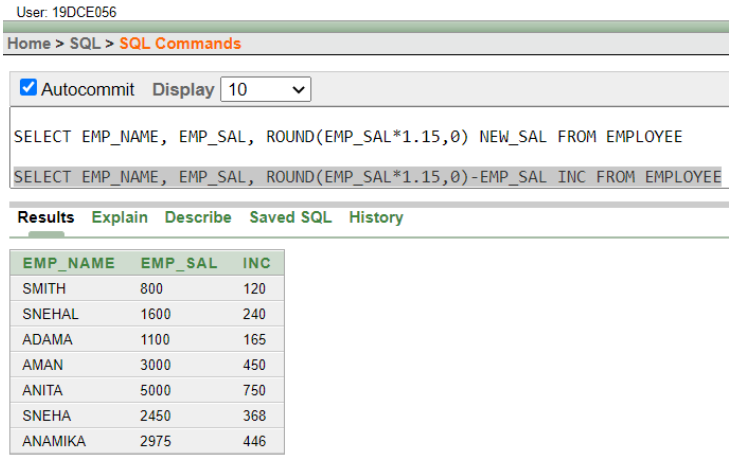


SELECT EMP_NAME, EMP_SAL, ROUND(EMP_SAL*1.15,0) "NEW_SAL" FROM EMPLOYEE



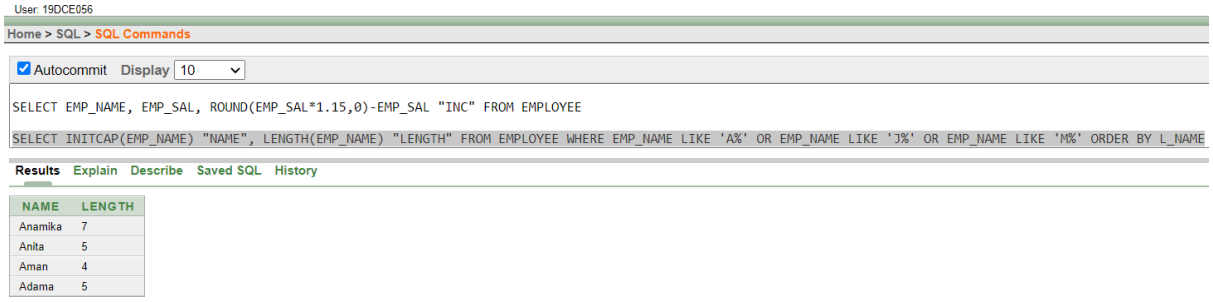
EMP_NAME	EMP_SAL	NEW_SAL
SMITH	800	920
SNEHAL	1600	1840
ADAMA	1100	1265
AMAN	3000	3450
ANITA	5000	5750
SNEHA	2450	2818
ANAMIKA	2975	3421

SELECT EMP_NAME, EMP_SAL, ROUND(EMP_SAL*1.15,0)-EMP_SAL "INC" FROM EMPLOYEE



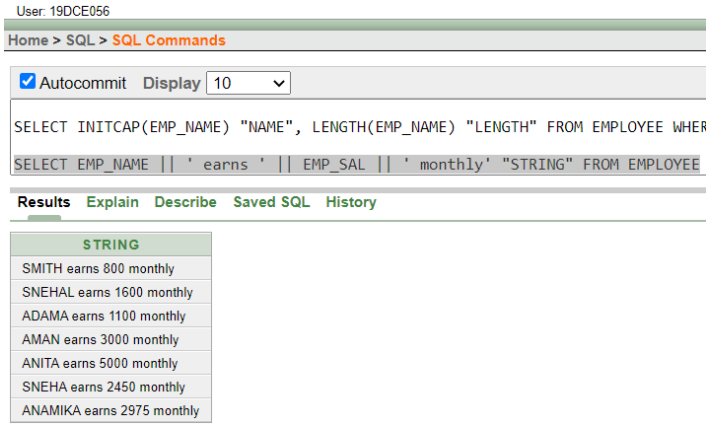
EMP_NAME	EMP_SAL	INC
SMITH	800	120
SNEHAL	1600	240
ADAMA	1100	165
AMAN	3000	450
ANITA	5000	750
SNEHA	2450	368
ANAMIKA	2975	446

SELECT INITCAP(EMP_NAME) "NAME", LENGTH(EMP_NAME) "LENGTH" FROM EMPLOYEE WHERE EMP_NAME LIKE 'A%' OR EMP_NAME LIKE 'J%' OR EMP_NAME LIKE 'M%' ORDER BY L_NAME



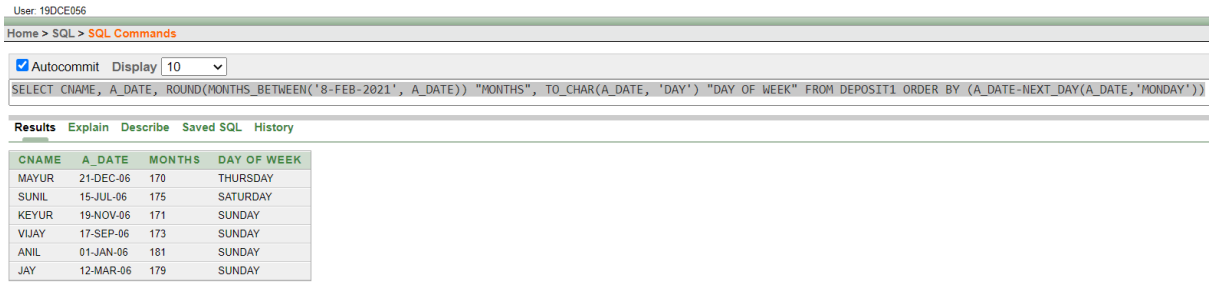
NAME	LENGTH
Anamika	7
Anita	5
Aman	4
Adama	5

SELECT EMP_NAME || ' earns ' || EMP_SAL || ' monthly' "STRING" FROM EMPLOYEE



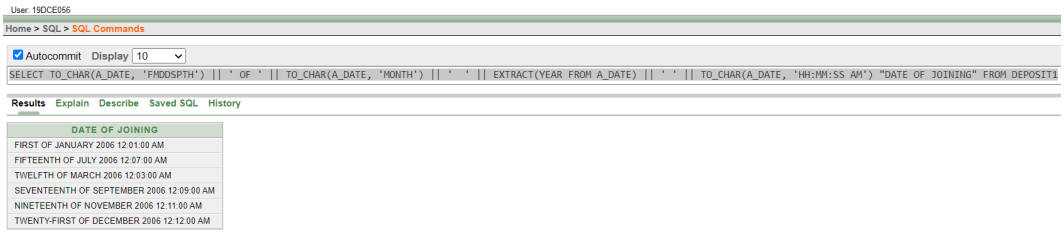
STRING
SMITH earns 800 monthly
SNEHAL earns 1600 monthly
ADAMA earns 1100 monthly
AMAN earns 3000 monthly
ANITA earns 5000 monthly
SNEHA earns 2450 monthly
ANAMIKA earns 2975 monthly

SELECT CNAME, A_DATE, ROUND(MONTHS_BETWEEN('8-FEB-2021', A_DATE)) "MONTHS", TO_CHAR(A_DATE, 'DAY') "DAY OF WEEK" FROM DEPOSIT1 ORDER BY (A_DATE-NEXT_DAY(A_DATE,'MONDAY'))



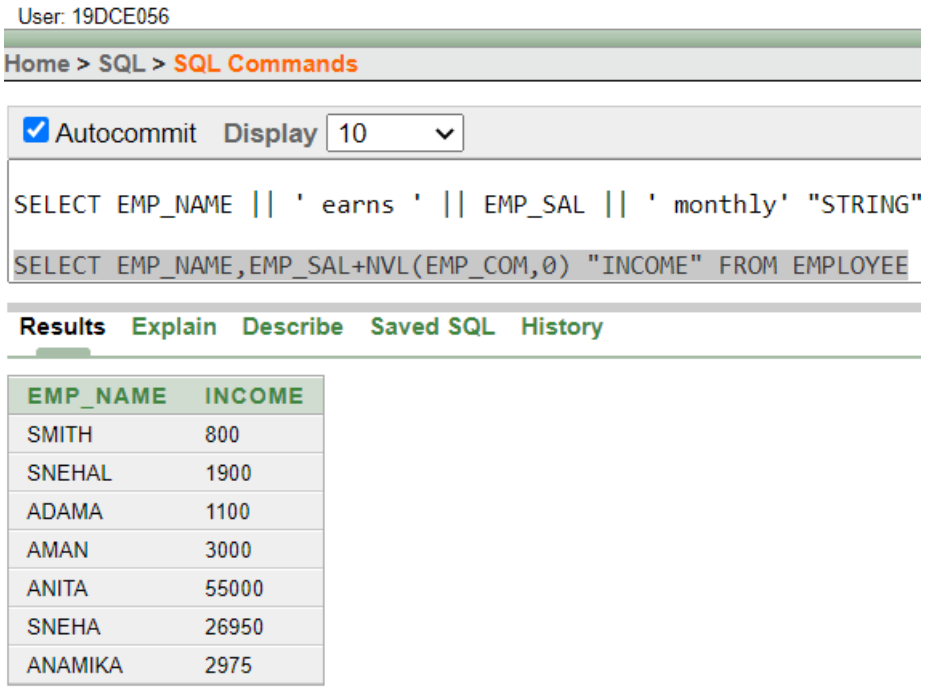
CNAME	A_DATE	MONTHS	DAY OF WEEK
MAYUR	21-DEC-06	170	THURSDAY
SUNIL	15-JUL-06	175	SATURDAY
KEYUR	19-NOV-06	171	SUNDAY
VIJAY	17-SEP-06	173	SUNDAY
ANIL	01-JAN-06	181	SUNDAY
JAY	12-MAR-06	179	SUNDAY

SELECT TO_CHAR(A_DATE, 'FMDDSPTH') || ' OF ' || TO_CHAR(A_DATE, 'MONTH') || ' ' || EXTRACT(YEAR FROM A_DATE) || ' ' || TO_CHAR(A_DATE, 'HH:MM:SS AM') "DATE OF JOINING" FROM DEPOSIT1



DATE OF JOINING
FIRST OF JANUARY 2006 12:01:00 AM
FIFTEENTH OF JULY 2006 12:07:00 AM
TWELFTH OF MARCH 2006 12:03:00 AM
SEVENTEENTH OF SEPTEMBER 2006 12:09:00 AM
NINETEENTH OF NOVEMBER 2006 12:11:00 AM
TWENTY-FIRST OF DECEMBER 2006 12:12:00 AM

SELECT EMP_NAME,EMP_SAL+NVL(EMP_COM,0) "INCOME" FROM EMPLOYEE



EMP_NAME	INCOME
SMITH	800
SNEHAL	1900
ADAMA	1100
AMAN	3000
ANITA	55000
SNEHA	26950
ANAMIKA	2975

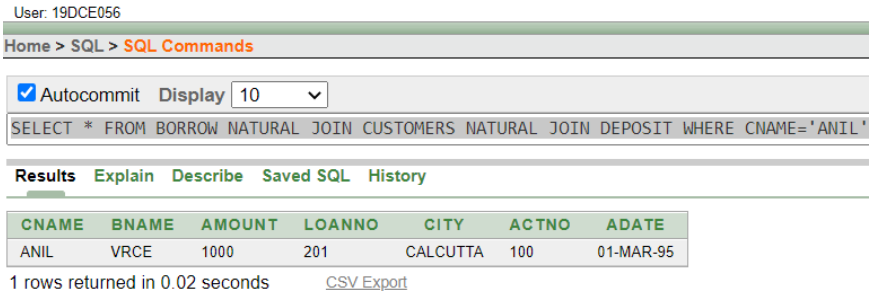
CONCLUSION: After performing this practical, we understood single-row functions in SQL such as TO_CHAR, ROUND, INITCAP etc.

PRACTICAL – 7

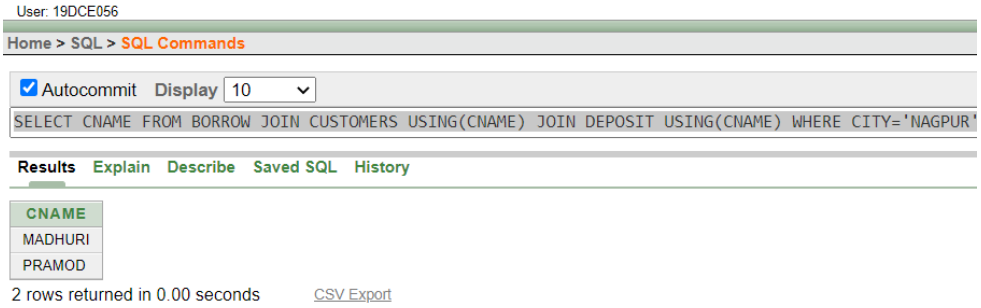
AIM: Displaying data from Multiple Tables (join)

COMMANDS:

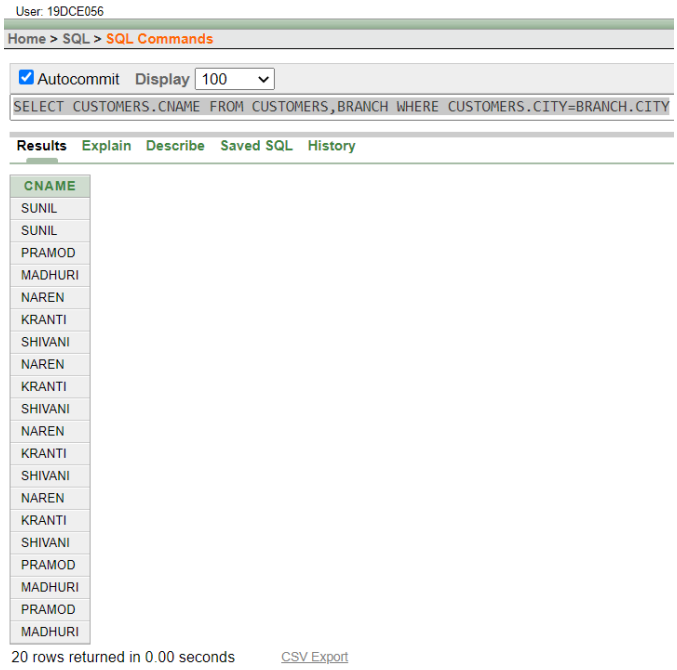
SELECT * FROM BORROW NATURAL JOIN CUSTOMERS NATURAL JOIN DEPOSIT WHERE CNAME='ANIL'



SELECT CNAME FROM BORROW JOIN CUSTOMERS USING(CNAME) JOIN DEPOSIT USING(CNAME) WHERE CITY='NAGPUR'



SELECT CUSTOMERS.CNAME FROM CUSTOMERS,BRANCH WHERE CUSTOMERS.CITY=BRANCH.CITY



SELECT DEPT_NO,L_NAME,DEPT_NAME FROM EMPLOYEE

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 100

```
SELECT DEPT_NO,L_NAME,DEPT_NAME FROM EMPLOYEE
```

Results Explain Describe Saved SQL History

DEPT_NO	L_NAME	DEPT_NAME
20	SHAH	MACHINE LEARNING
25	GUPTA	DATA SCIENCE
20	WALES	MACHINE LEARNING
15	SHARMA	VIRUAL REALITY
10	PATEL	BIG DATA ANALYTICS
10	JOSEPH	BIG DATA ANALYTICS
30	JHA	ARTIFICIAL INTELLEGENCE

SELECT * FROM JOB JOIN EMPLOYEE USING(JOB_ID) WHERE DEPT_NO=30

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 100

```
SELECT * FROM JOB JOIN EMPLOYEE USING(JOB_ID) WHERE DEPT_NO=30
```

Results Explain Describe Saved SQL History

JOB_ID	JOB_TITLE	MIN_SAL	MAX_SAL	EMP_NO	EMP_NAME	EMP_SAL	EMP_COM	DEPT_NO	L_NAME	DEPT_NAME	LOCATION	MANAGER_ID	HIREDATE
IT_PROG	Programmer	4000	10000	107	ANAMIKA	2975	-	30	JHA	ARTIFICIAL INTELLEGENCE	NEW YORK	-	15-JUL-97

SELECT EMP_NAME,DEPT_NO,DEPT_NAME FROM EMPLOYEE WHERE LOCATION='NEW YORK'

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 100

```
SELECT EMP_NAME,DEPT_NO,DEPT_NAME FROM EMPLOYEE WHERE LOCATION='NEW YORK'
```

Results Explain Describe Saved SQL History

EMP_NAME	DEPT_NO	DEPT_NAME
ANAMIKA	30	ARTIFICIAL INTELLEGENCE

SELECT E1.EMP_NO,E1.L_NAME,E2.EMP_NO,E2.L_NAME FROM EMPLOYEE E1 JOIN EMPLOYEE E2 ON E1.MANAGER_ID=E2.EMP_NO

User: 19DCE056

Home > SQL > SQL Commands

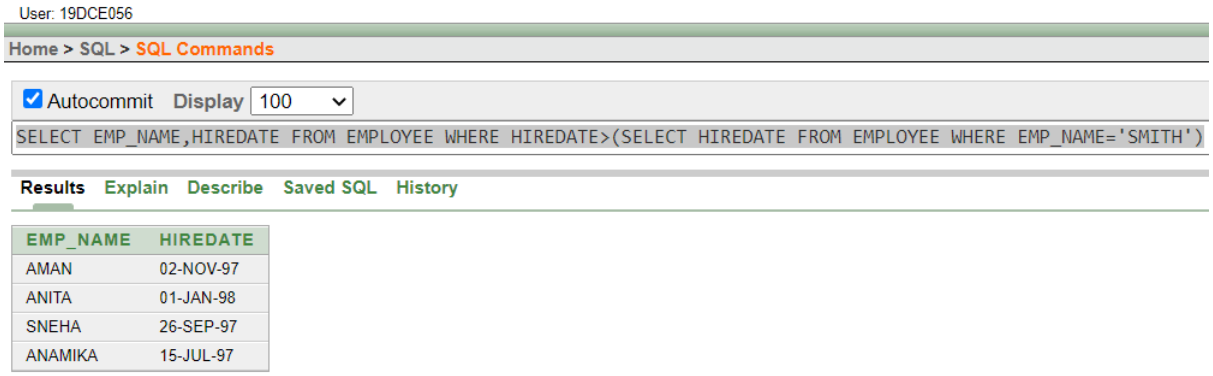
Autocommit Display 100

```
SELECT E1.EMP_NO,E1.L_NAME,E2.EMP_NO,E2.L_NAME FROM EMPLOYEE E1 JOIN EMPLOYEE E2 ON E1.MANAGER_ID=E2.EMP_NO
```

Results Explain Describe Saved SQL History

EMP_NO	L_NAME	EMP_NO	L_NAME
106	JOSEPH	105	PATEL
103	WALES	105	PATEL
101	SHAH	105	PATEL
105	PATEL	107	JHA

SELECT EMP_NAME,HIREDATE FROM EMPLOYEE WHERE HIREDATE> (SELECT HIREDATE FROM EMPLOYEE WHERE EMP_NAME='SMITH')



EMP_NAME	HIREDATE
AMAN	02-NOV-97
ANITA	01-JAN-98
SNEHA	26-SEP-97
ANAMIKA	15-JUL-97

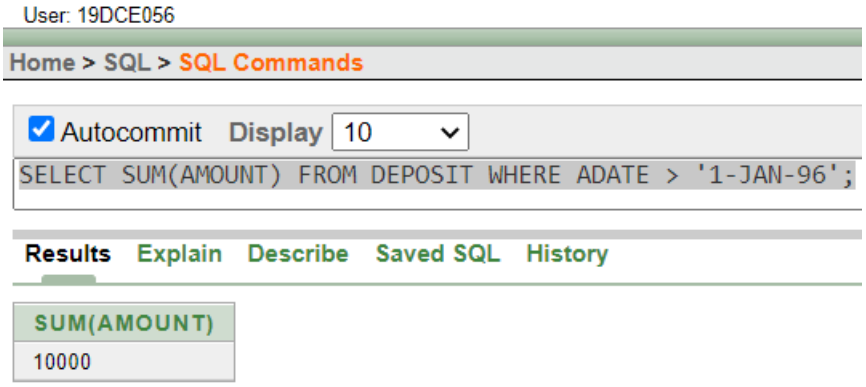
CONCLUSION: After performing this practical, we understood various types of JOIN in SQL.

PRACTICAL – 8

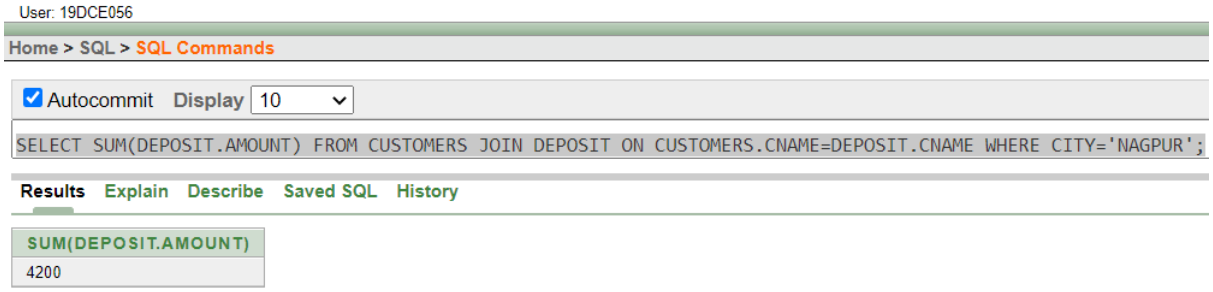
AIM: To apply the concept of Aggregating Data using Group functions.

COMMANDS:

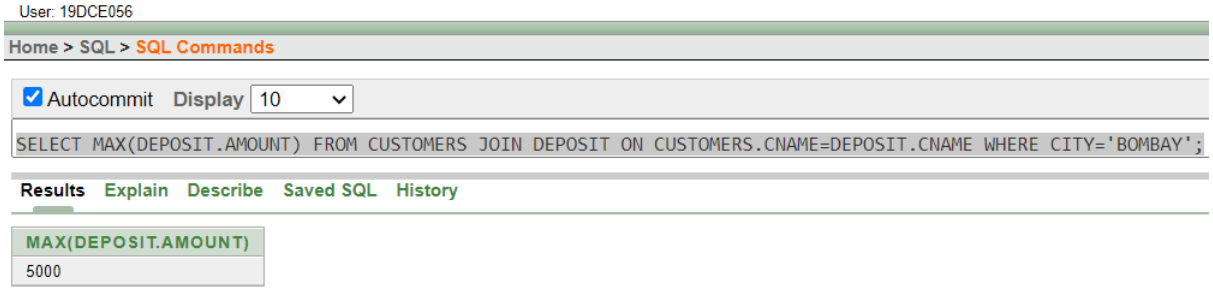
SELECT SUM(AMOUNT) FROM DEPOSIT WHERE ADATE > '1-JAN-96';



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



SELECT MAX(DEPOSIT.AMOUNT) FROM CUSTOMERS JOIN DEPOSIT ON CUSTOMERS.CNAME=DEPOSIT.CNAME WHERE CITY='BOMBAY';



SELECT ROUND (AVG(EMP_SAL)) "Average", MAX(EMP_SAL) "Maximum" , MIN(EMP_SAL) "Minimum", SUM(EMP_SAL) "Sum" FROM EMPLOYEE;

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT ROUND (AVG(EMP_SAL)) "Average", MAX(EMP_SAL) "Maximum" , MIN(EMP_SAL) "Minimum", SUM(EMP_SAL) "Sum" FROM EMPLOYEE;
```

Results Explain Describe Saved SQL History

Average	Maximum	Minimum	Sum
2418	5000	800	16925

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

User: 19DCE056

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

SELECT TO_CHAR(HIREDATE, 'YYYY'), COUNT(EMP_NO) FROM EMPLOYEE GROUP BY TO_CHAR(HIREDATE, 'YYYY')

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT TO_CHAR(HIREDATE, 'YYYY'), COUNT(EMP_NO) FROM EMPLOYEE GROUP BY TO_CHAR(HIREDATE, 'YYYY')
```

Results Explain Describe Saved SQL History

TO_CHAR(HIREDATE,'YYYY')	COUNT(EMP_NO)
1997	3
1995	1
1996	2
1998	1

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

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

Results Explain Describe Saved SQL History

AVG(EMP_SAL)	DEPT_NO
1600	25
2975	30
950	20
3000	15
3725	10

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

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

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

Results Explain Describe Saved SQL History

DEPT_NO	JOB_ID	SUM(EMP_SAL)
20	MK_MGR	1100
15	COMP_OP	3000
30	IT_PROG	2975
20	FI_MGR	800
10	COMP_OP	5000
25	LEC	1600
10	FI_ACC	2450

SELECT ROUND(AVG(EMP_SAL)) FROM EMPLOYEE GROUP BY DEPT_NO HAVING AVG(EMP_SAL) > 2000

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT ROUND(AVG(EMP_SAL)) FROM EMPLOYEE GROUP BY DEPT_NO HAVING AVG(EMP_SAL) > 2000
```

Results Explain Describe Saved SQL History

ROUND(AVG(EMP_SAL))
2975
3000
3725

SELECT JOB_ID, SUM(EMP_SAL) FROM EMPLOYEE GROUP BY JOB_ID HAVING SUM(EMP_SAL) > 3000 ORDER BY SUM(EMP_SAL)

User: 19DCE056

Home > SQL > SQL Commands

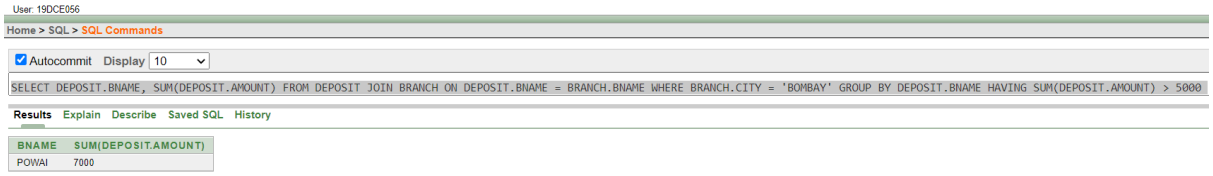
Autocommit Display 10

```
SELECT JOB_ID, SUM(EMP_SAL) FROM EMPLOYEE GROUP BY JOB_ID HAVING SUM(EMP_SAL) > 3000 ORDER BY SUM(EMP_SAL)
```

Results Explain Describe Saved SQL History

JOB_ID	SUM(EMP_SAL)
COMP_OP	8000

SELECT DEPOSIT.BNAME, SUM(DEPOSIT.AMOUNT) FROM DEPOSIT JOIN BRANCH ON DEPOSIT.BNAME = BRANCH.BNAME WHERE BRANCH.CITY = 'BOMBAY' GROUP BY DEPOSIT.BNAME HAVING SUM(DEPOSIT.AMOUNT) > 5000



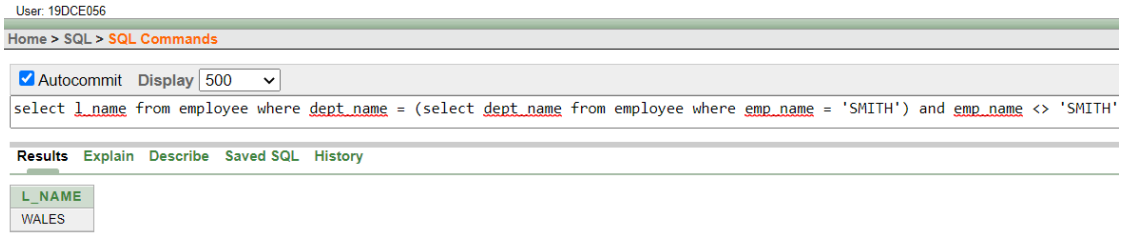
CONCLUSION: After performing this practical, we understood the concept of group and various aggregate functions.

PRACTICAL – 9

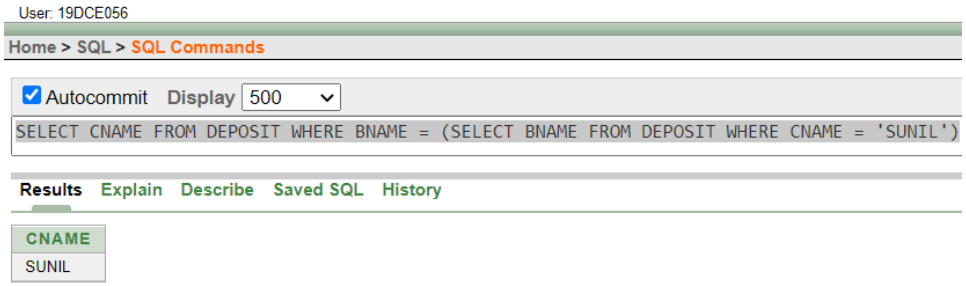
AIM: To solve queries using the concept of sub query.

COMMANDS:

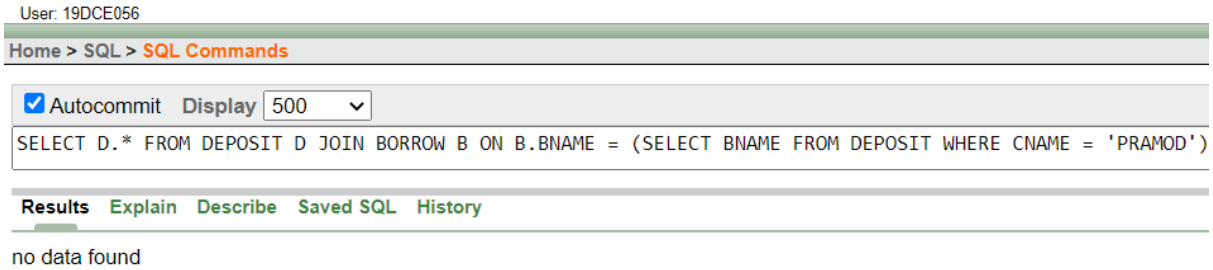
select l_name from employee where dept_name = (select dept_name from employee where emp_name = 'SMITH') and emp_name <> 'SMITH'



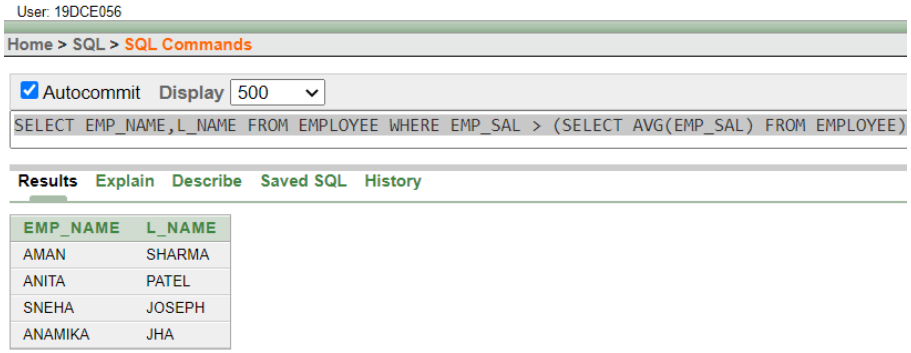
SELECT CNAME FROM DEPOSIT WHERE BNAME = (SELECT BNAME FROM DEPOSIT WHERE CNAME = 'SUNIL')



SELECT D.* FROM DEPOSIT D JOIN BORROW B ON B.BNAME = (SELECT BNAME FROM DEPOSIT WHERE CNAME = 'PRAMOD')



SELECT EMP_NAME,L_NAME FROM EMPLOYEE WHERE EMP_SAL > (SELECT AVG(EMP_SAL) FROM EMPLOYEE)



SELECT CNAME FROM DEPOSIT WHERE BNAME = (SELECT BNAME FROM DEPOSIT WHERE CNAME = 'ANIL') AND AMOUNT > 2000

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 500

```
SELECT CNAME FROM DEPOSIT WHERE BNAME = (SELECT BNAME FROM DEPOSIT WHERE CNAME = 'ANIL') AND AMOUNT > 2000
```

Results Explain Describe Saved SQL History

no data found

SELECT L_NAME,EMP_SAL FROM EMPLOYEE WHERE MANAGER_ID = (SELECT EMP_NO FROM EMPLOYEE WHERE EMP_NAME = 'Ford')

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 500

```
SELECT L_NAME,EMP_SAL FROM EMPLOYEE WHERE MANAGER_ID = (SELECT EMP_NO FROM EMPLOYEE WHERE EMP_NAME = 'Ford')
```

Results Explain Describe Saved SQL History

no data found

SELECT E.DEPT_NAME,E.EMP_NAME,E.JOB_ID FROM EMPLOYEE E WHERE E.JOB_ID = (SELECT JOB_ID FROM JOB WHERE JOB_TITLE = 'Account')

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT E.DEPT_NAME,E.EMP_NAME,E.JOB_ID FROM EMPLOYEE E WHERE E.JOB_ID = (SELECT JOB_ID FROM JOB WHERE JOB_TITLE = 'Account')
```

Results Explain Describe Saved SQL History

DEPT_NAME	EMP_NAME	JOB_ID
BIG DATA ANALYTICS	SNEHA	FI_ACC

SELECT D1.BNAME FROM DEPOSIT D1 GROUP BY D1.BNAME HAVING COUNT(D1.BNAME) >= ALL(SELECT COUNT(D2.BNAME) FROM DEPOSIT D2 GROUP BY D2.BNAME)

User: 19DCE056

Home > SQL > SQL Commands

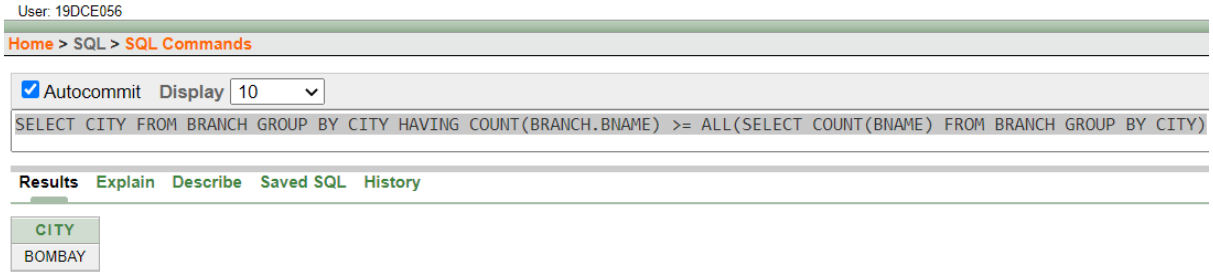
Autocommit Display 10

```
SELECT D1.BNAME FROM DEPOSIT D1 GROUP BY D1.BNAME HAVING COUNT(D1.BNAME) >= ALL(SELECT COUNT(D2.BNAME) FROM DEPOSIT D2 GROUP BY D2.BNAME)
```

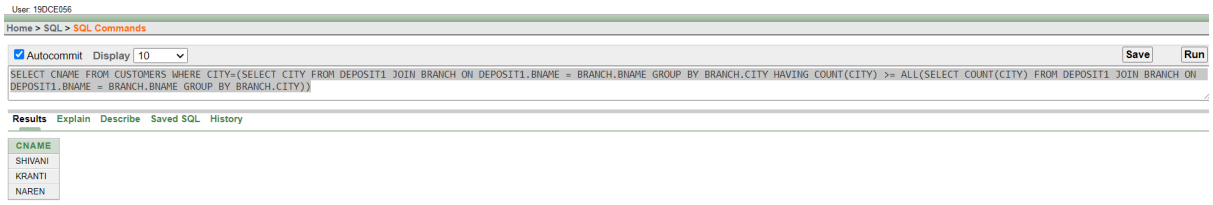
Results Explain Describe Saved SQL History

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

SELECT CITY FROM BRANCH GROUP BY CITY HAVING COUNT(BRANCH.BNAME) >= ALL(SELECT COUNT(BNAME) FROM BRANCH GROUP BY CITY)



SELECT CNAME FROM CUSTOMERS WHERE CITY=(SELECT CITY FROM DEPOSIT1 JOIN BRANCH ON DEPOSIT1.BNAME = BRANCH.BNAME GROUP BY BRANCH.CITY HAVING COUNT(CITY) >= ALL(SELECT COUNT(CITY) FROM DEPOSIT1 JOIN BRANCH ON DEPOSIT1.BNAME = BRANCH.BNAME GROUP BY BRANCH.CITY))

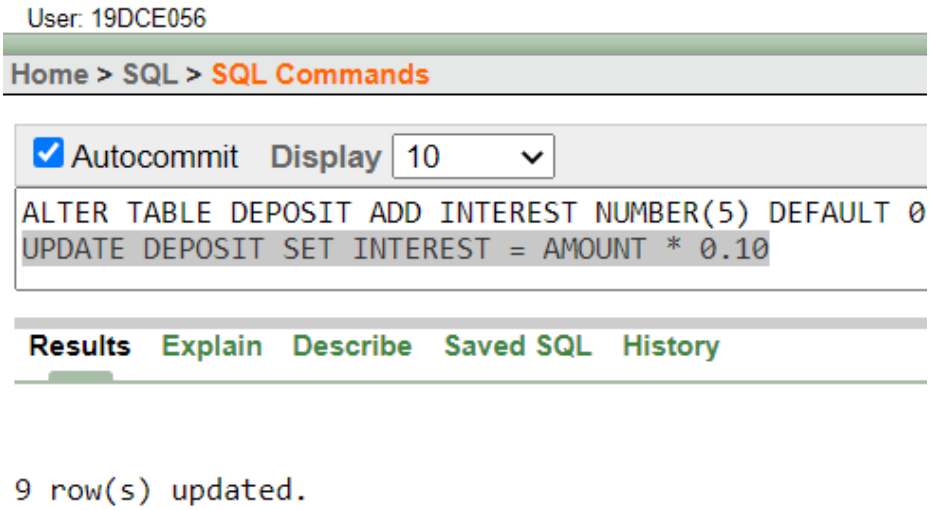


CONCLUSION: After performing this practical, we understood the concept of sub query.

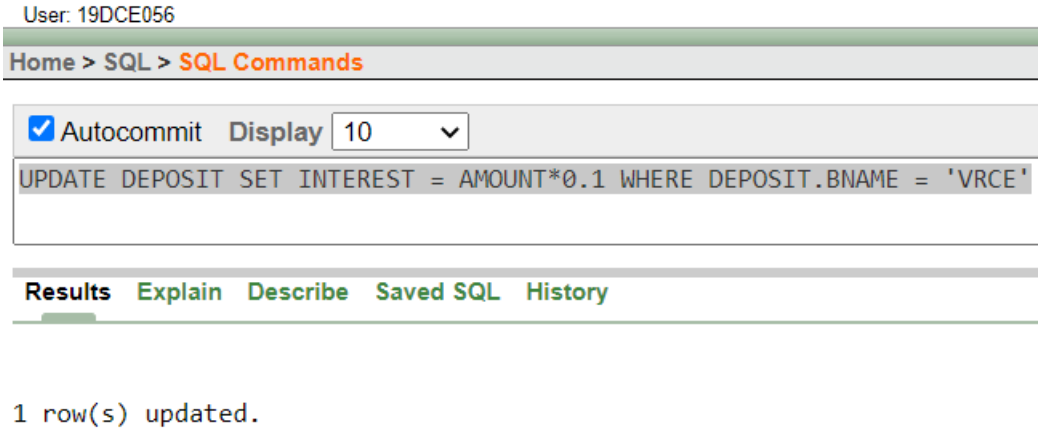
PRACTICAL – 10

AIM: To solve queries using the concept of Data Manipulation.

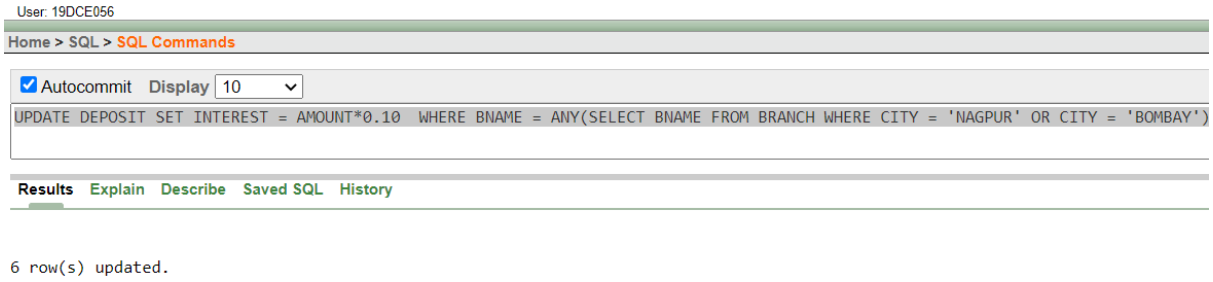
COMMANDS:
ALTER TABLE DEPOSIT ADD INTEREST NUMBER(5) DEFAULT 0
UPDATE DEPOSIT SET INTEREST = AMOUNT * 0.10



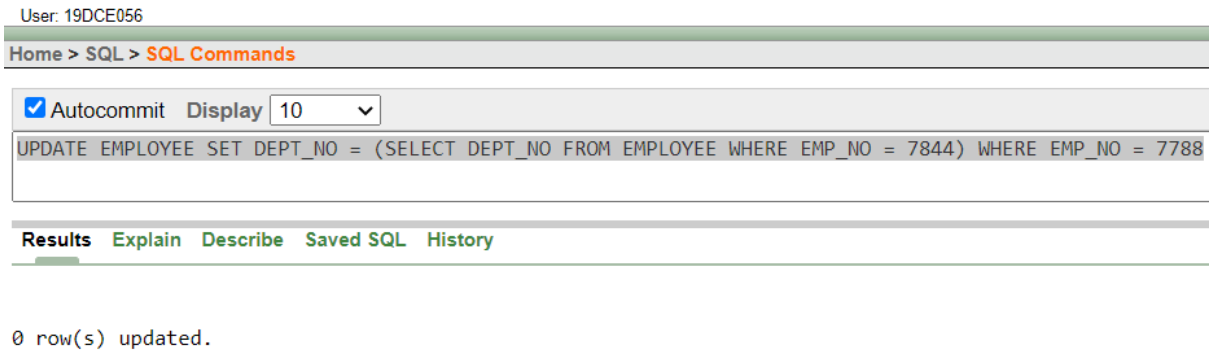
UPDATE DEPOSIT SET INTEREST = AMOUNT*0.1 WHERE DEPOSIT.BNAME = 'VRCE'



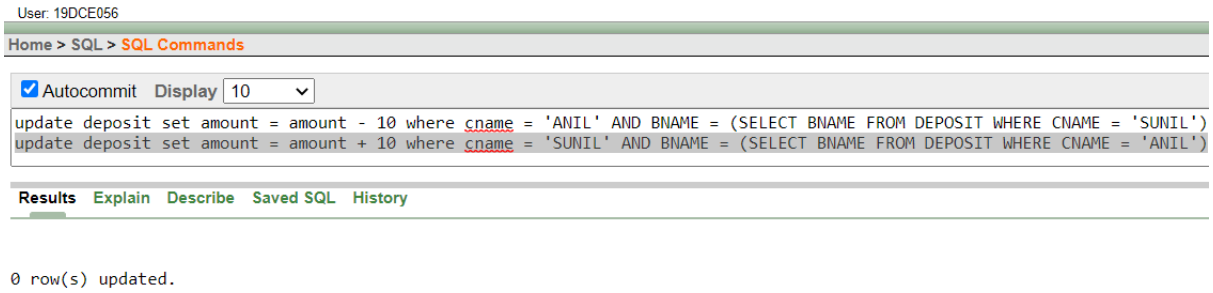
UPDATE DEPOSIT SET INTEREST = AMOUNT*0.10 WHERE BNAME = ANY(SELECT BNAME FROM BRANCH WHERE CITY = 'NAGPUR' OR CITY = 'BOMBAY')



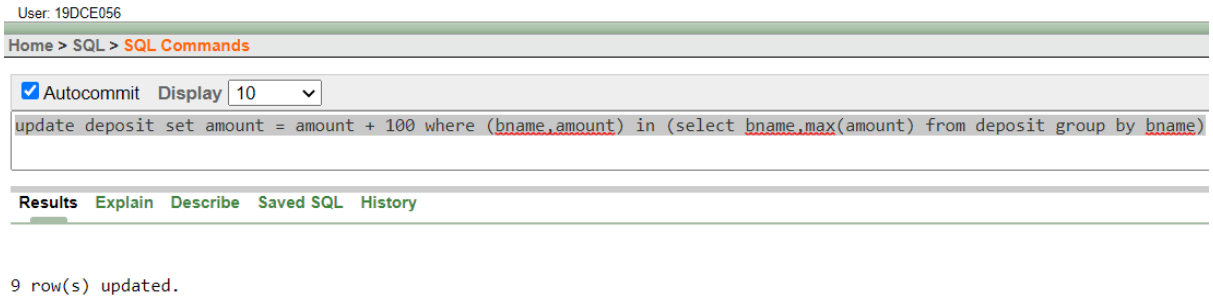
UPDATE EMPLOYEE SET DEPT_NO = (SELECT DEPT_NO FROM EMPLOYEE WHERE EMP_NO = 7844) WHERE EMP_NO = 7788



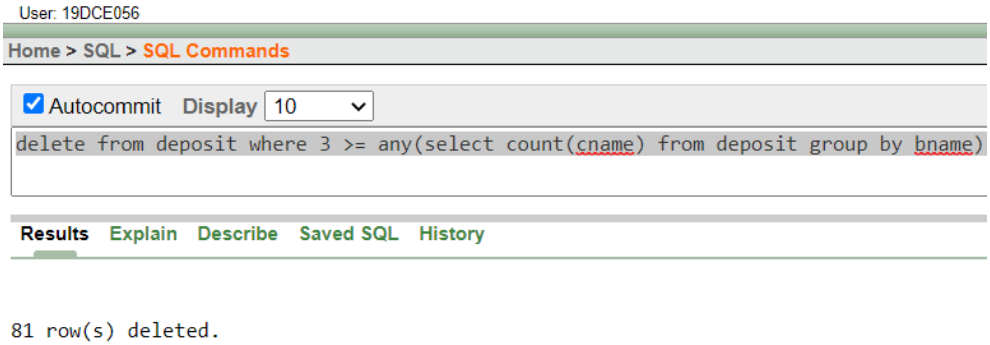
update deposit set amount = amount - 10 where cname = 'ANIL' AND BNAME = (SELECT BNAME FROM DEPOSIT WHERE CNAME = 'SUNIL')
update deposit set amount = amount + 10 where cname = 'SUNIL' AND BNAME = (SELECT BNAME FROM DEPOSIT WHERE CNAME = 'ANIL')



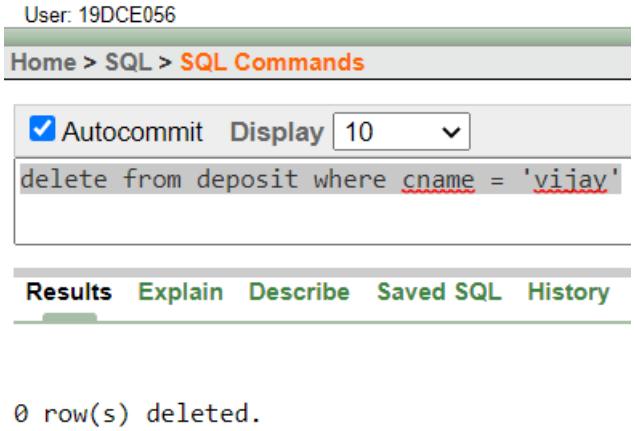
update deposit set amount = amount + 100 where (bname,amount) in (select bname,max(amount) from deposit group by bname)



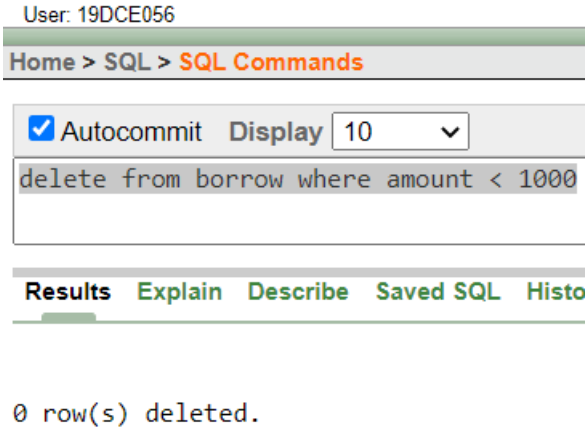
delete from deposit where 3 >= any(select count(cname) from deposit group by bname)



delete from deposit where cname = 'vijay'



delete from borrow where amount < 1000



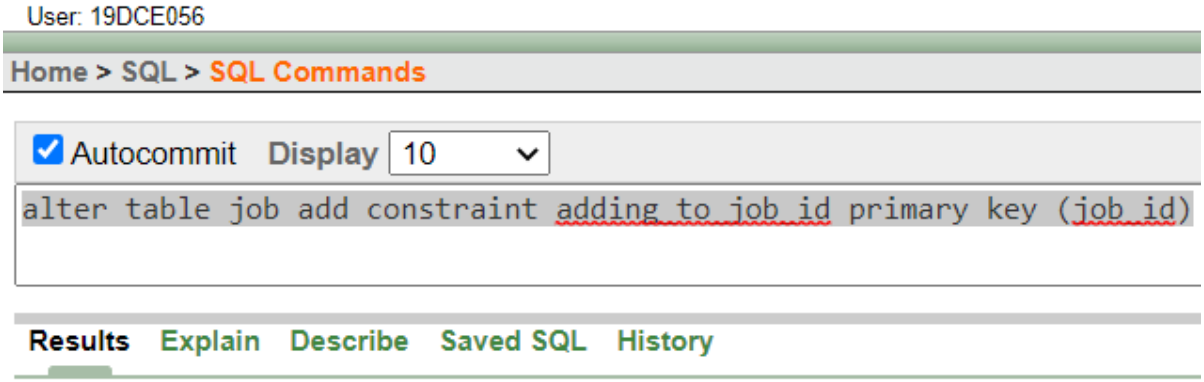
CONCLUSION: After performing this practical, we understood the DML part of DBMS.

PRACTICAL – 11

AIM: To solve queries using the concept of constraints.

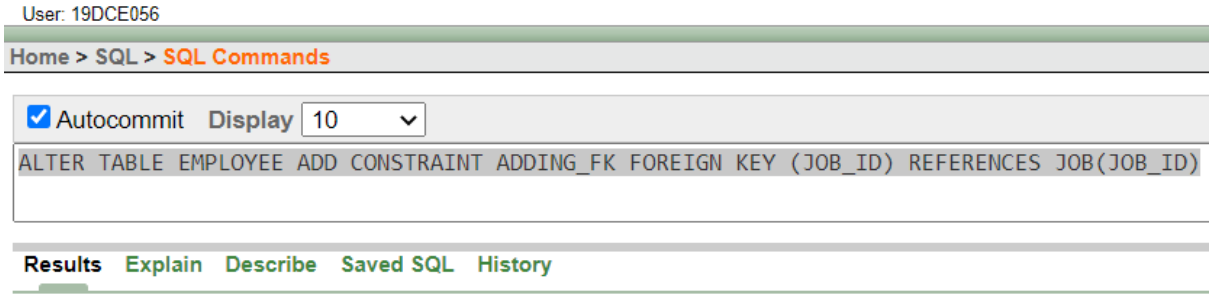
COMMANDS:

alter table job add constraint adding_to_job_id primary key (job_id)



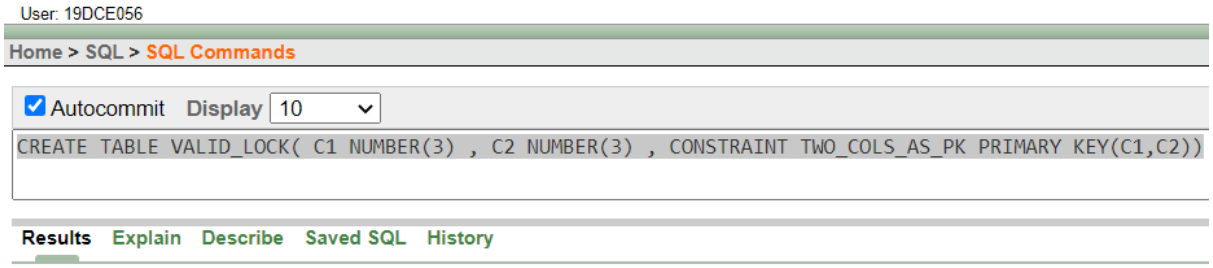

```
Table altered.
```

ALTER TABLE EMPLOYEE ADD CONSTRAINT ADDING_FK FOREIGN KEY (JOB_ID) REFERENCES JOB(JOB_ID)



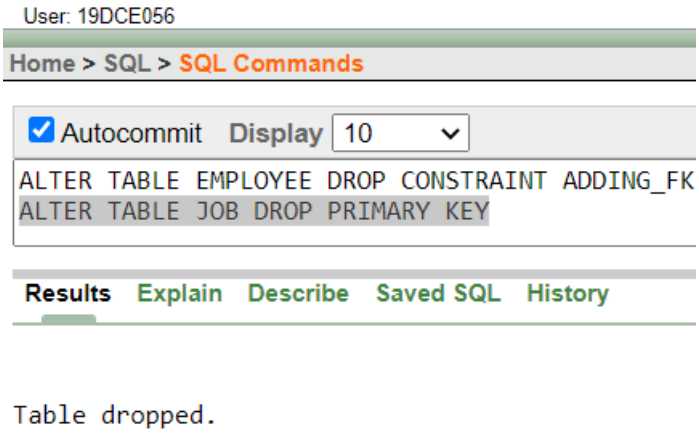

```
Table altered.
```

CREATE TABLE VALID_LOCK(C1 NUMBER(3) , C2 NUMBER(3) , CONSTRAINT TWO_COLS_AS_PK PRIMARY KEY(C1,C2))




```
Table created.
```

5. ALTER TABLE EMPLOYEE DROP CONSTRAINT ADDING_FK
4. ALTER TABLE JOB DROP PRIMARY KEY



CONCLUSION: After performing this practical, we understood the concept of constraints.

PRACTICAL – 12

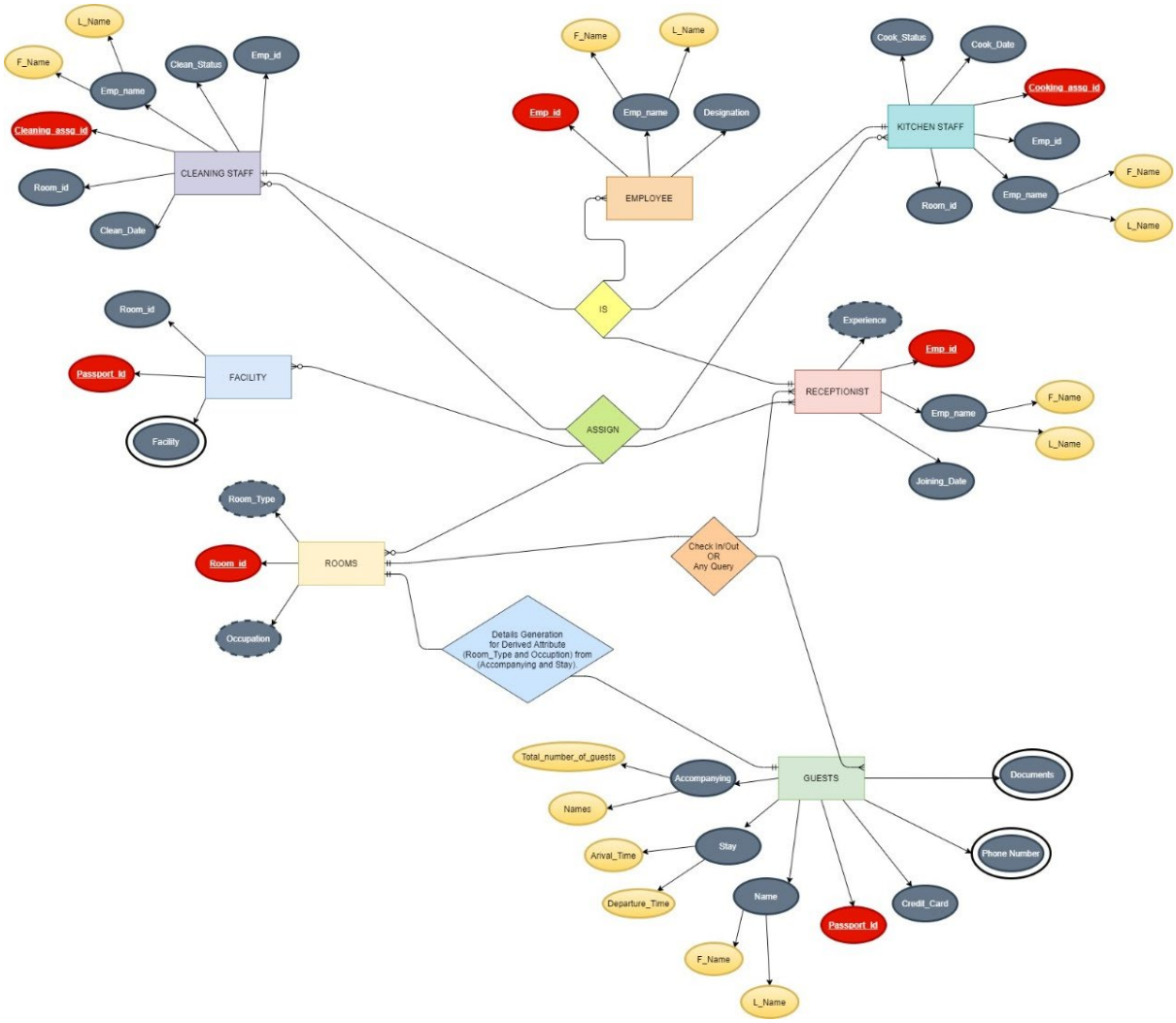
AIM: Data Dictionary and E-R Diagram

COMMANDS:

DATA DICTIONARY

1	Field Name	Field Name	Field Name	Field Name	Example	Constraints
2	emp_id	number	6	Employee Unique Id	100120	PK
3	emp_name	Text	20	Employee Name (F_Name+L_Name)	Jeet Undaviya	Non Null
4	F_Name	Text	10	First Name	Jeet	Non Null
5	L_Name	Text	10	Last Name	Undaviya	Non Null
6	Designation	Text	10	Designation of an employee	Cleaner	Non Null
7	Joining_Date	Date	8	Date of Joinng	12-03-2006	Non Null
8	year_of_experience	number	2	Current year - Joining year	5 Years	Non Null
9	Cleaning_assig_id	Text	16	Format :- Date(6)+Emp_id(6)+Room_id(4) This is Unique assigned id work by cleaning staff.	12032006100120101	PK
10	Room_id	number	4	Unique Room Number	1011	PK
11	Room_Type	Text	2	It can be either SR(Single Room),DR(Double Room) or TR(Triple Room). Also it is Derived Attribute (Depending on the number of persons accompning guests).	SR	Non Null
12	Occupation	Date/Time	10	Room will be occupied (Departure Time - Arrival Time). And it is also derived attribute.	48 Hours	Non Null
13	Credit_Card	number	16	Credit Card Number and its details.	1234 4567 8901	Unique And Not Null
14	Documents	Text	50	More details about guest like their address proof, driving licence or adhar card.	Drving Licence :- kadjakcd1243	Non Null
15	Passport_id	number	9	PassPort Number of guest.	123456789	PK
16	PhoneNumber	number	10	Phone number of guest. There can more then one number	623456789	Unique And Not Null
17	Arrival_Time	Date/Time	10	Arrival date with time.	01/01/2007 12:30pm	Non Null
18	Depurture_Time	Date/Time	10	Depature date with time.	02/01/2007 12:30pm	Non Null
19	Name	Text	20	Guest Name (F_Name +L_Name).	Meet Sheth	Non Null
20	Accompaning	Text	30	Other people accompanying guest.It is contains of total number of guest and names of them.If guest is single it should be filled with "None".	total guest :- 12 ;Other Guests Names:- Yug,Tanmay,Shrey,Jay,Deep, Vaibhav,Darsham,Priyanshu, Yash,Jaimin Don	Non Null
21	Clean_Date	Date/Time	10	Cleaning Date and time	01/01/2007 12:00pm	Non Null
22	Clean_Status	Text	10	Its decribes current cleaning status of an assigned room.	Cleaning Done.	Non Null
23	Facility	Text	20	It can have mutiple activities like(Gym and Swimming) according to guest package.	Gym,GameZone and Swimming	Non Null
24	Cooking_assign_id	Text	16	Format :- Date(6)+Emp_id(6)+Room_id(4) This is Unique assigned id work by cooking staff.	12032006100120101	PK
25	Cook_Date	Date/Time	10	Cooking Date and Time	01/01/2007 2:00pm	Non Null
26	Cook_Status	Text	10	Its decribes current cooking order status given for the guest for assigned room.	Food is ready.	Non Null
27	Stay	Date/Time	10	Guest Stay (Departure Time - Arrival Time). And it is also derived attribute.	48 Hours	Non Null

ER Diagram



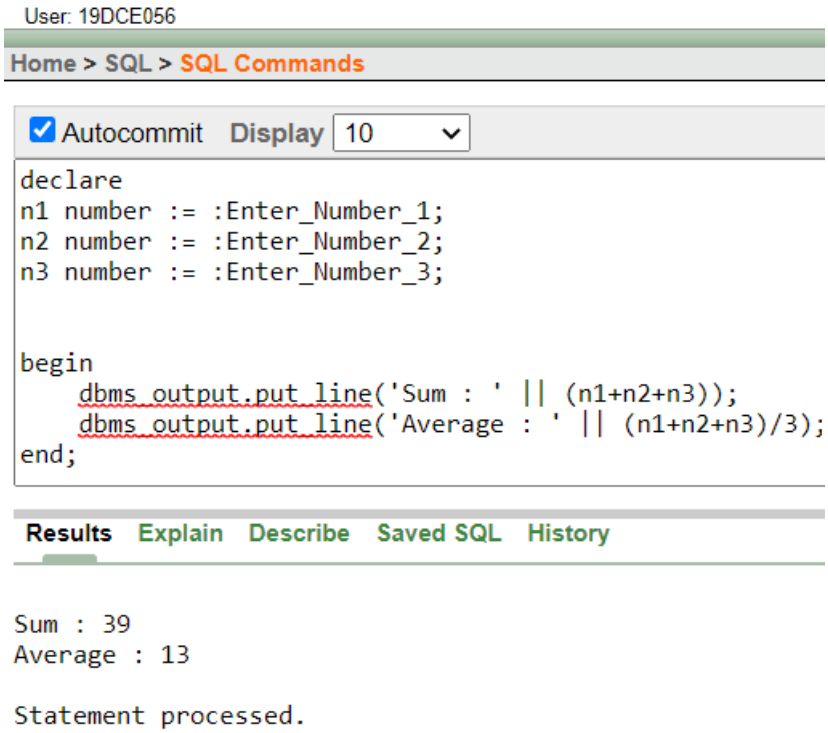
CONCLUSION: After performing this practical, we understood the concepts like data dictionary and ER-diagrams.

PRACTICAL – 13

AIM: Write a PL-SQL block to find Sum and average of three numbers.

COMMANDS:

```
declare
n1 number := :Enter_Number_1;
n2 number := :Enter_Number_2;
n3 number := :Enter_Number_3;


begin
   dbms_output.put_line('Sum : ' || (n1+n2+n3));
   dbms_output.put_line('Average : ' || (n1+n2+n3)/3);
end;
```



CONCLUSION: After Performing this practical, we understood the basic concepts of PL/SQL.

PRACTICAL – 14

AIM: Find the factorial of a number in pl/sql using for, While and Simple Loop.

COMMANDS:

```
declare
n1 number := :Enter_Number;
fact number := 1;
temp number;

begin
   temp := n1;

   loop
      fact := fact*temp;
      temp := temp-1;
      if temp=1 then
         exit;
      end if;
   end loop;

   dbms_output.put_line('Factorial : ' || fact);

   fact := 1;
   temp := n1;
   while temp>0 loop
      fact := fact*temp;
      temp := temp-1;
   end loop;

   dbms_output.put_line('Factorial : ' || fact);

   fact := 1;
   for i in 1..n1 loop
      fact := fact*i;
   end loop;

   dbms_output.put_line('Factorial : ' || fact);
end;
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
    dbms_output.put_line('Factorial : ' || fact);

    fact := 1;
for i in 1..n1 loop
        fact := fact*i;
    end loop;

    dbms_output.put_line('Factorial : ' || fact);
end;
```

Results Explain Describe Saved SQL History

```
Factorial : 479001600
Factorial : 479001600
Factorial : 479001600

Statement processed.
```

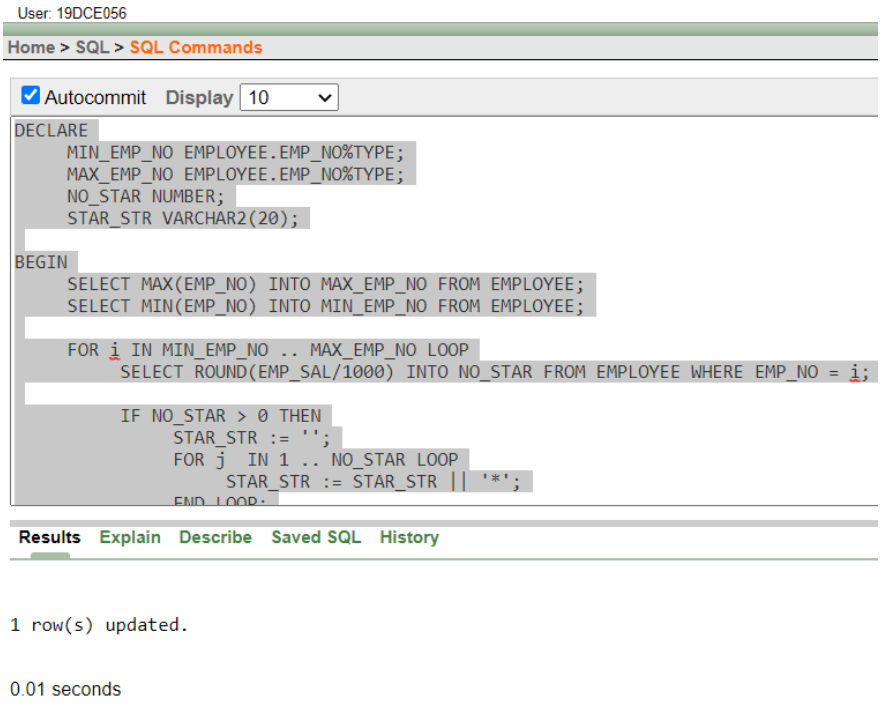
CONCLUSION: After Performing this practical, we understood the concept of Loop in PL/SQL.

PRACTICAL – 15

AIM: To understand the concept of "select into" and "% type" attribute.

COMMANDS:

```
DECLARE
    MIN_EMP_NO EMPLOYEE.EMP_NO%TYPE;
    MAX_EMP_NO EMPLOYEE.EMP_NO%TYPE;
    NO_STAR NUMBER;
    STAR_STR VARCHAR2(20);

BEGIN
    SELECT MAX(EMP_NO) INTO MAX_EMP_NO FROM EMPLOYEE;
    SELECT MIN(EMP_NO) INTO MIN_EMP_NO FROM EMPLOYEE;
    FOR i IN MIN_EMP_NO .. MAX_EMP_NO LOOP
        SELECT ROUND(EMP_SAL/1000) INTO NO_STAR FROM EMPLOYEE WHERE
EMP_NO = i;

        IF NO_STAR > 0 THEN
            STAR_STR := '';
            FOR j  IN 1 .. NO_STAR LOOP
                STAR_STR := STAR_STR || '*';
            END LOOP;
        END IF;
        UPDATE EMPLOYEE SET STARS = STAR_STR WHERE EMP_NO = i ;
    END LOOP;
END;
```



CONCLUSION: After performing this practical, we understood the concept of **select into** and **%type** in PL/SQL.

PRACTICAL – 16

AIM: To understand the concept of cursor.

COMMANDS:
(A)
```
DECLARE
CURSOR GETALLDETAILS IS SELECT * FROM EMPLOYEE;
VAR EMPLOYEE%ROWTYPE;

BEGIN
   dbms_output.put_line('ALL DETAILS ARE AS FOLLOWS :- ');
   for I in GETALLDETAILS LOOP
      VAR := I;
      dbms_output.put_line(VAR.EMP_NO||'  '||  VAR.EMP_NAME||'  '||VAR.EMP_SAL||'
'||VAR.EMP_COM||'    '||VAR.DEPT_NO||'    '||VAR.L_NAME||'    '||VAR.DEPT_NAME||'
'||VAR.JOB_ID||' '||VAR.LOCATION||' '||VAR.MANAGER_ID||' '||VAR.HIREDATE);
   END LOOP;
END;
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DECLARE
CURSOR GETALLDETAILS IS SELECT * FROM EMPLOYEE;
VAR EMPLOYEE%ROWTYPE;

BEGIN
    dbms_output.put_line('ALL DETAILS ARE AS FOLLOWS :- ');
    for I in GETALLDETAILS LOOP
        VAR := I;
        dbms_output.put_line(VAR.EMP_NO||' '|| VAR.EMP_NAME||' '||VAR.EMP_SAL||'
'||VAR.MANAGER_ID||' '||VAR.HIREDATE);
    END LOOP;
END;
```

Results Explain Describe Saved SQL History

```
ALL DETAILS ARE AS FOLLOWS :-
101 SMITH 800  20 SHAH MACHINE LEARNING FI_MGR TORONTO 105 09-AUG-96
102 SNEHAL 1600 300 25 GUPTA DATA SCIENCE LEC LAS VEGAS  14-MAR-96
103 ADAMA 1100 0 20 WALES MACHINE LEARNING MK_MGR ONTARIO 105 30-NOV-95
104 AMAN 3000  15 SHARMA VIRUAL REALITY COMP_OP MEXICO 12 02-NOV-97
105 ANITA 5000 50000 10 PATEL BIG DATA ANALYTICS COMP_OP GERMANY 107 01-JAN-98
106 SNEHA 2450 24500 10 JOSEPH BIG DATA ANALYTICS FI_ACC MELBOURNE 105 26-SEP-97
107 ANAMIKA 2975  30 JHA ARTIFICIAL INTELLEGENCE IT_PROG NEW YORK  15-JUL-97

Statement processed.
```

(B)

```
DECLARE
DEPT_NUMBER NUMBER := :ENTER_DEPT_NO_;
CURSOR GETBYDEPT IS SELECT EMP_NAME,L_NAME, EMP_SAL, MANAGER_ID
FROM EMPLOYEE WHERE DEPT_NO = DEPT_NUMBER;

BEGIN
   FOR I IN GETBYDEPT LOOP
      IF I.EMP_SAL < 1000 AND (I.MANAGER_ID = 7902 OR I.MANAGER_ID = 7839)
THEN
         DBMS_OUTPUT.PUT_LINE(I.L_NAME || ', DUE FOR A RAISE.');
      ELSE
         DBMS_OUTPUT.PUT_LINE(I.L_NAME || ', NOT DUE FOR A RAISE.');
      END IF;
   END LOOP;
END;
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
DECLARE
DEPT_NUMBER NUMBER := :ENTER_DEPT_NO_;
CURSOR GETBYDEPT IS SELECT EMP_NAME,L_NAME, EMP_SAL, MANAGER_ID FROM EMPLOYEE WHERE DEPT_NO = DEPT_NUMBER;

BEGIN
    FOR I IN GETBYDEPT LOOP
        IF I.EMP_SAL < 1000 AND (I.MANAGER_ID = 7902 OR I.MANAGER_ID = 7839) THEN
            DBMS_OUTPUT.PUT_LINE(I.L_NAME || ', DUE FOR A RAISE.');
        ELSE
            DBMS_OUTPUT.PUT_LINE(I.L_NAME || ', NOT DUE FOR A RAISE.');
        END IF;
    END LOOP;
END;
```

Results Explain Describe Saved SQL History

```
Statement processed.
```

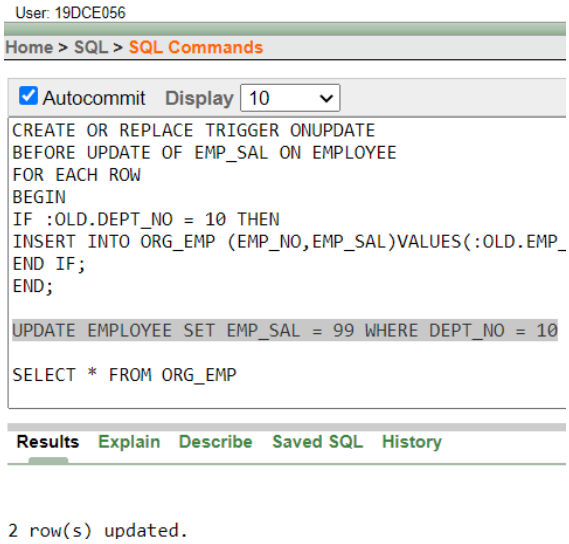
CONCLUSION: After performing this practical, we understood the concept of cursor in PL/SQL.

PRACTICAL – 17

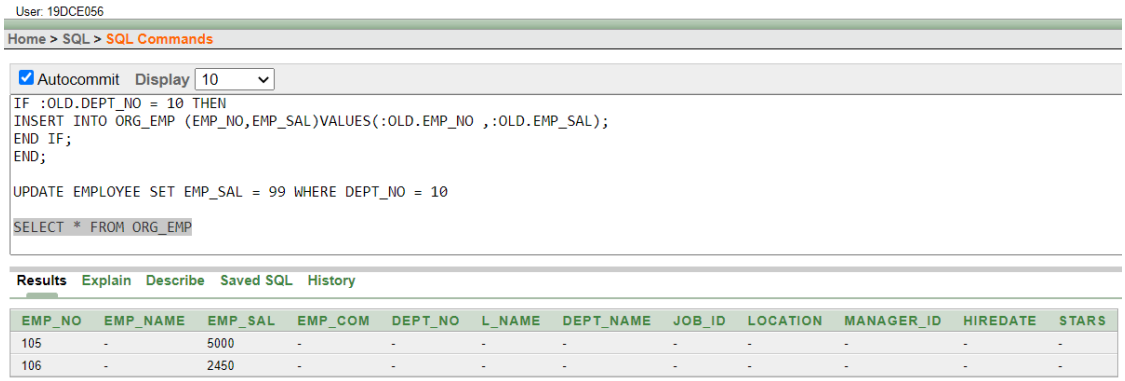
AIM: To perform the concept of trigger.

COMMANDS:

```
CREATE TABLE ORG_EMP AS SELECT * FROM EMPLOYEE WHERE EMP_NAME = 'NO_NAME'

CREATE OR REPLACE TRIGGER ONUPDATE
BEFORE UPDATE OF EMP_SAL ON EMPLOYEE
FOR EACH ROW
BEGIN
IF :OLD.DEPT_NO = 10 THEN
INSERT INTO ORG_EMP (EMP_NO,EMP_SAL)VALUES(:OLD.EMP_NO ,:OLD.EMP_SAL);
END IF;
END;
UPDATE EMPLOYEE SET EMP_SAL = 99 WHERE DEPT_NO = 10
```




```
SELECT * FROM ORG_EMP
```



EMP_NO	EMP_NAME	EMP_SAL	EMP_COM	DEPT_NO	L_NAME	DEPT_NAME	JOB_ID	LOCATION	MANAGER_ID	HIREDATE	STARS
105	-	5000	-	-	-	-	-	-	-	-	-
106	-	2450	-	-	-	-	-	-	-	-	-

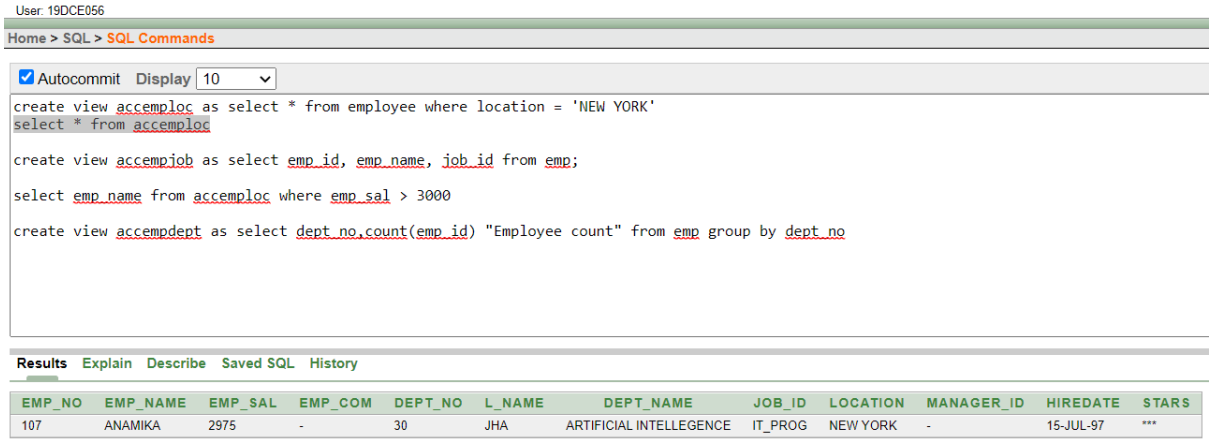
CONCLUSION: After performing this practical, we understood the concept of trigger in PL/SQL.

PRACTICAL – 18

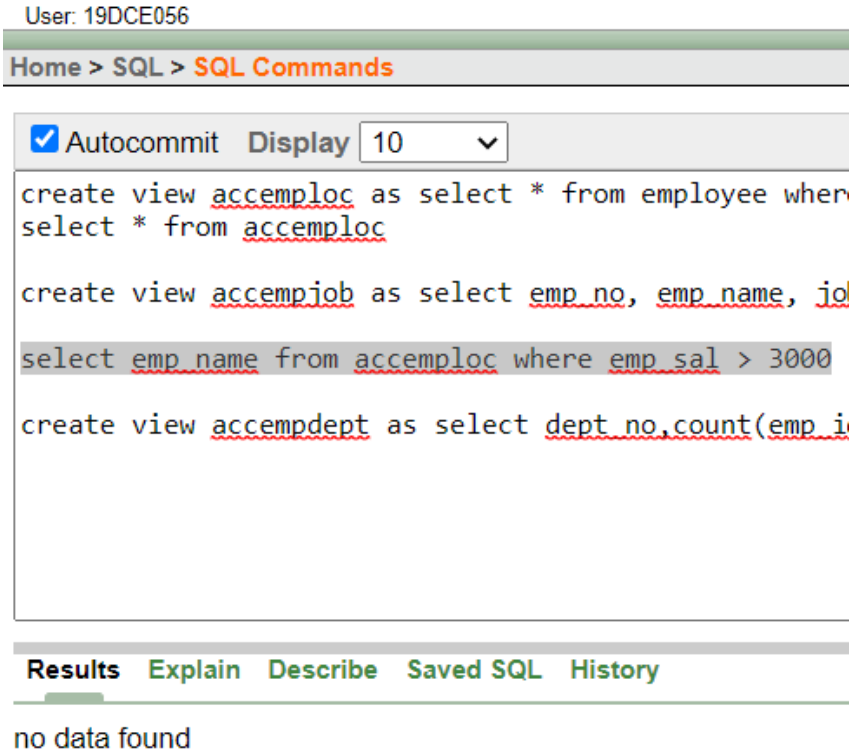
AIM: To solve queries using concept of view.

COMMANDS:

create view accemploc as select * from employee where location = 'NEW YORK'
select * from accemploc



create view accempjob as select emp_no, emp_name, job_id from employee;
select emp_name from accemploc where emp_sal > 3000



create view accempdept as select dept_no,count(emp_no) "Employee count" from employee group by dept_no

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
create view accemploc as select * from employee where location = 'NEW YORK'
select * from accemploc

create view accempjob as select emp_no, emp_name, job_id from employee;

select emp_name from accemploc where emp_sal > 3000

create view accempdept as select dept_no,count(emp_no) "Employee count" from employee group by dept_no
```

Results Explain Describe Saved SQL History

```
View created.
```

CONCLUSION: After performing this practical, we understood the concept of view in PL/SQL.

PRACTICAL – 19

To perform the concept of function and procedure.

COMMANDS:

```
create or replace procedure update_emp_sal(empid number,newsal number) is
row_updated number;
begin
update employee set emp_sal = newsal where emp_no = empid;
row_updated := SQL%rowcount;
if row_updated = 0 then
dbms_output.put_line('Salary not updated ! No employee with Emp_no '||empid||' found !');
else
dbms_output.put_line('Salary updated for employee with Emp_no '||empid||'.');
end if;
end;
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
create or replace procedure update_emp_sal(empid number,newsal number) is
row_updated number;
begin
update employee set emp_sal = newsal where emp_no = empid;
row_updated := SQL%rowcount;
if row_updated = 0 then
dbms_output.put_line('Salary not updated ! No employee with Emp_no '||empid||' found !');
else
dbms_output.put_line('Salary updated for employee with Emp_no '||empid||'.');
end if;
end;

create or replace procedure update_emp_sal(empid number,newsal number) is
```

Results Explain Describe Saved SQL History

```
Procedure created.
```


```
create or replace procedure update_emp_sal(empid number,newsal number) is
row_updated number;
begin
update employee set emp_sal = newsal where emp_no = empid;
row_updated := SQL%rowcount;
if row_updated = 0 then
dbms_output.put_line('Salary not updated ! No employee with Emp_no '||empid||' found !');
else
dbms_output.put_line('Salary updated for employee with Emp_no '||empid||'.');
end if;
end;
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
end if;
end;

create or replace procedure update_emp_sal(empid number,newsal number) is
row_updated number;
begin
update employee set emp_sal = newsal where emp_no = empid;
row_updated := SQL%rowcount;
if row_updated = 0 then
dbms_output.put_line('Salary not updated ! No employee with Emp_no '||empid||' found !');
else
dbms_output.put_line('Salary updated for employee with Emp_no '||empid||'.');
end if;
end;
```

Results Explain Describe Saved SQL History

```
Procedure created.
```

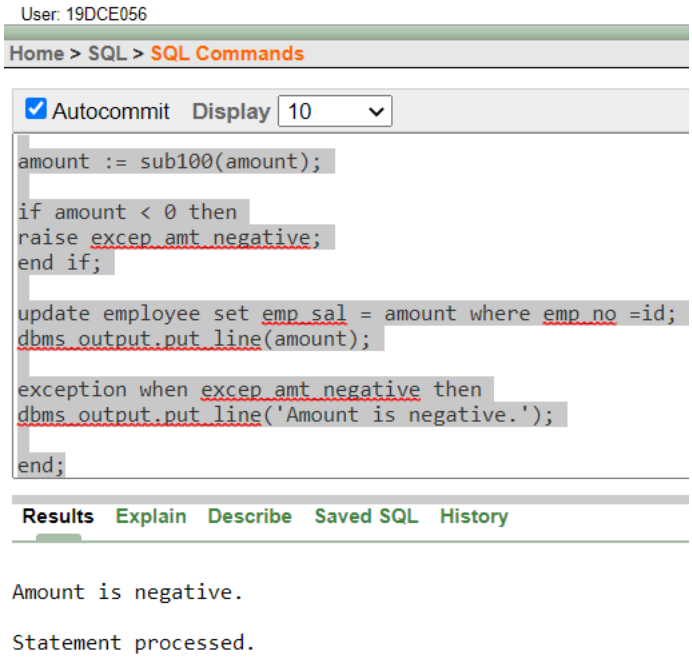
CONCLUSION: After performing this practical, we understood the concept of function and procedure in PL/SQL.

PRACTICAL – 20

To perform the concept of exception handler.

COMMANDS:

```
create or replace function add100(amount number) return number is
begin
return amount+100;
end;
create or replace function sub100(amount number) return number is
begin
return -100;
end;

declare
id number := :Enter_ID;
amount number := 0;
excep_amt_negative exception;
begin
select emp_sal into amount from employee where emp_no = id;
amount := sub100(amount);
if amount < 0 then
raise excep_amt_negative;
end if;
update employee set emp_sal = amount where emp_no =id;
dbms_output.put_line(amount);
exception when excep_amt_negative then
dbms_output.put_line('Amount is negative.');
end;
```



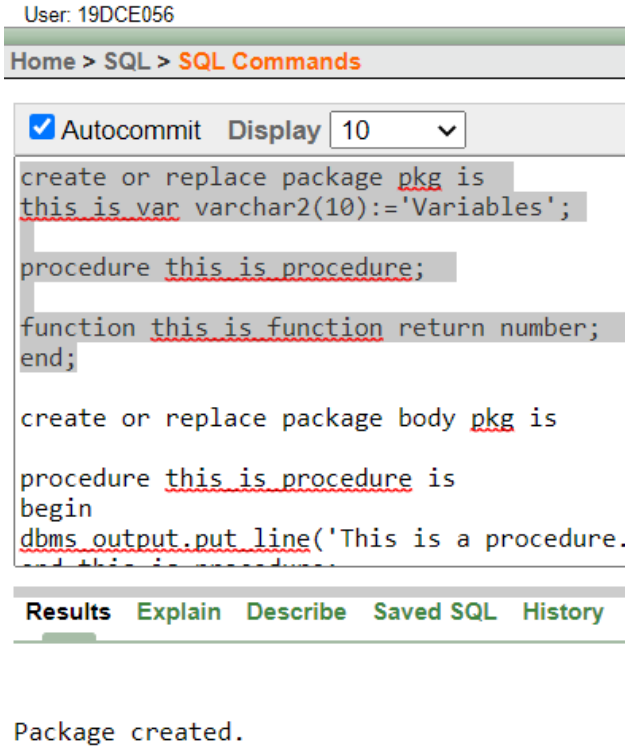
CONCLUSION: After performing this practical, we understood the concept of exception handler in PL/SQL.

PRACTICAL – 21

To perform the concept of package.

COMMANDS:

```
create or replace package pkg is
this_is_var varchar2(10):='Variables';

procedure this_is_procedure;

function this_is_function return number;
end;
```




```
create or replace package body pkg is

procedure this_is_procedure is
begin
dbms_output.put_line('This is a procedure.');
end this_is_procedure;

function this_is_function return number is
begin
dbms_output.put_line('This is a function returning 0.');
return 0;
end this_is_function;
end pkg;
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
begin
dbms_output.put_line('This is a function returning 0.');
return 0;
end this_is_function;
end pkg;

declare
flag number;
begin
dbms_output.put_line(pkg.this_is_var);
pkg.this_is_procedure();
flag := pkg.this_is_function();
dbms_output.put_line(flag);
end;
```

Results Explain Describe Saved SQL History

```
Package Body created.
```

```
declare
flag number;
begin
dbms_output.put_line(pkg.this_is_var);
pkg.this_is_procedure();
flag := pkg.this_is_function();
dbms_output.put_line(flag);
end;
```

User: 19DCE056

Home > SQL > SQL Commands

Autocommit Display 10

```
begin
dbms_output.put_line('This is a functi
return 0;
end this_is_function;
end pkg;

declare
flag number;
begin
dbms_output.put_line(pkg.this_is_var);
pkg.this_is_procedure();
flag := pkg.this_is_function();
dbms_output.put_line(flag);
end;
```

Results Explain Describe Saved SQL Histo

```
Variables
This is a procedure.
This is a function returning 0.
0

Statement processed.
```

CONCLUSION: After performing this practical, we understood the concept of package in PL/SQL.