

Practical -1

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

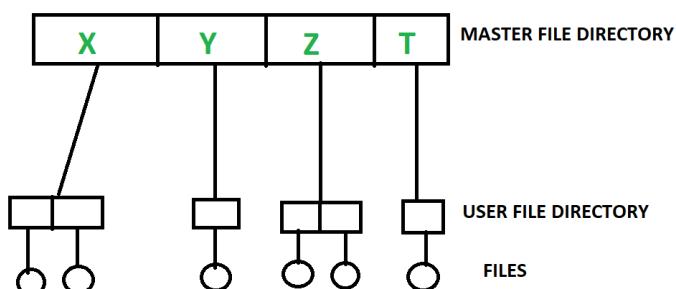
THEORY:

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).

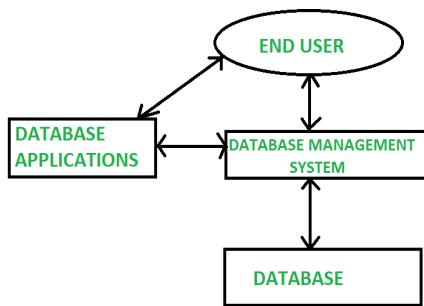


2. DBMS (Database Management System) :

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:

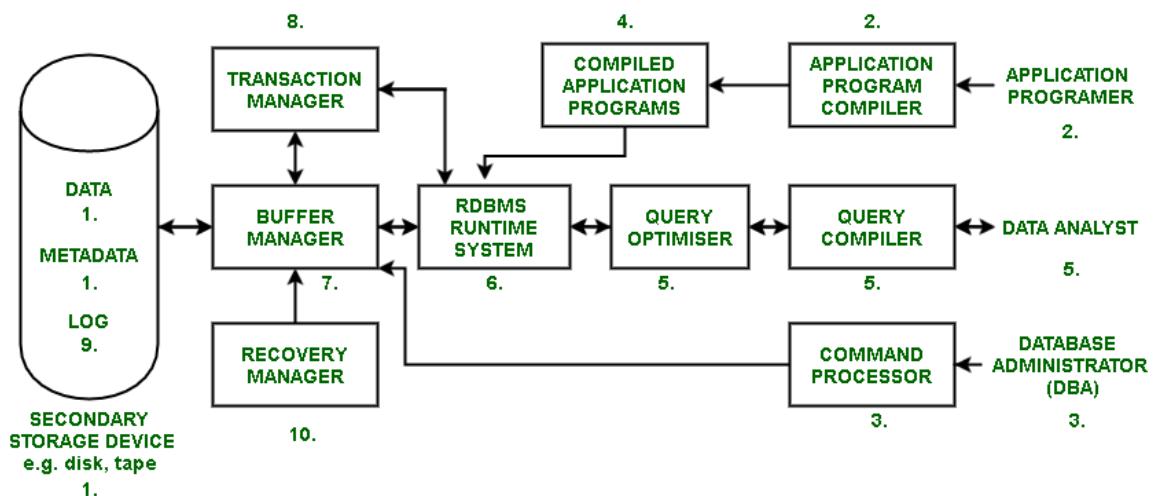
Oracle, MySQL, MS SQL server.



3. RDBMS:

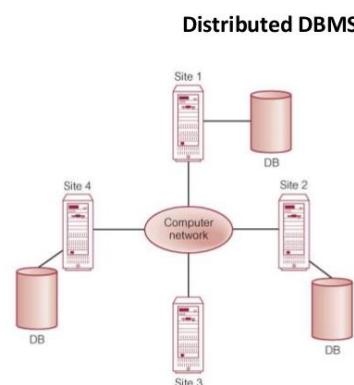
RDBMS stands for Relational Database Management System and it implements SQL. In the real-world scenario, people use the Relational Database Management System to collect information and process it, to provide service. E.g. In a ticket processing system, details about us (e.g. age, gender) and our journey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

RDBMS Architecture:



4. DDBMS

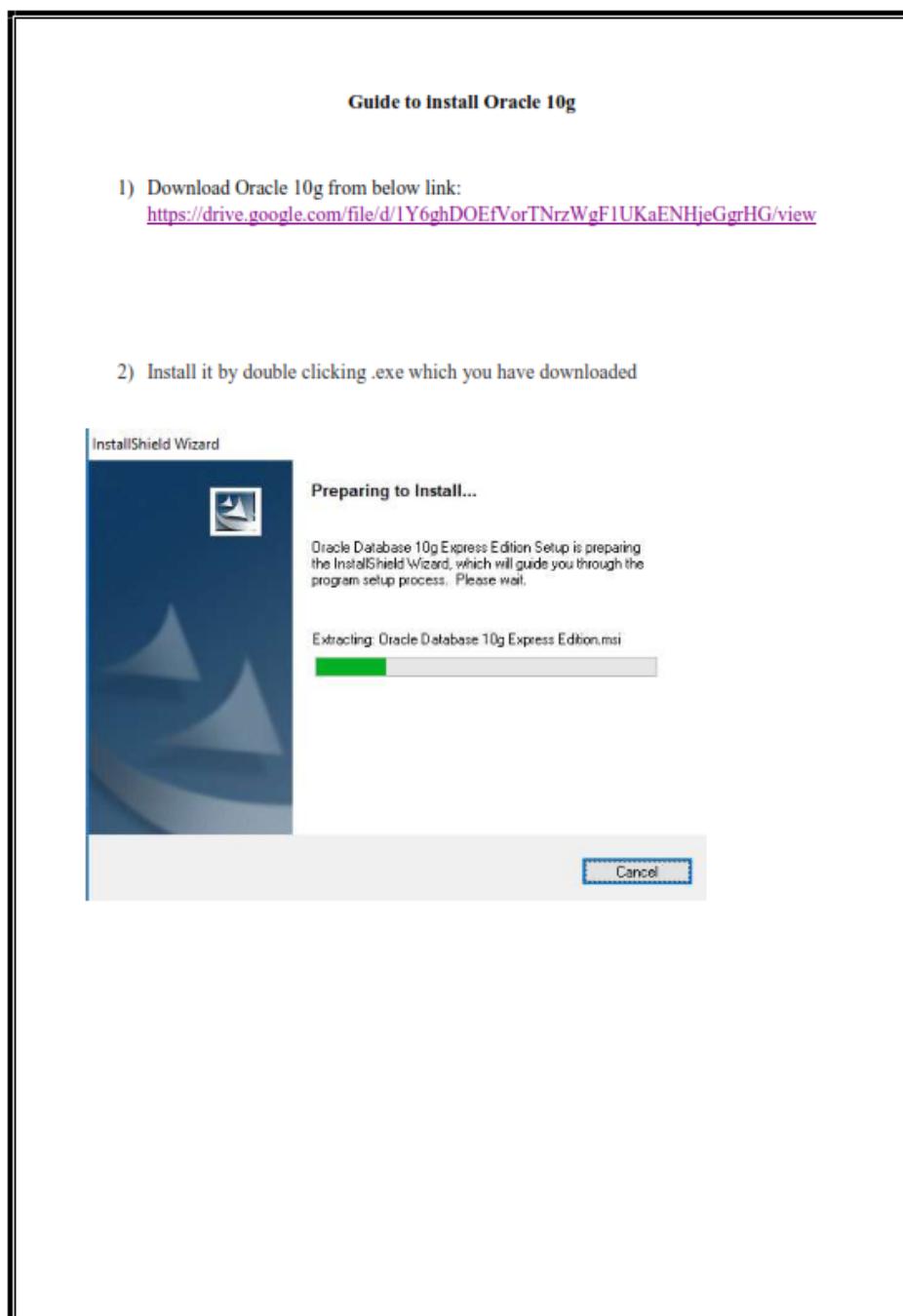
A distributed database is basically a database that is not limited to one system, it is spread over different sites, i.e., on multiple computers or over a network of computers. A distributed database system is located on various sites that don't share physical components. This may be required when a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



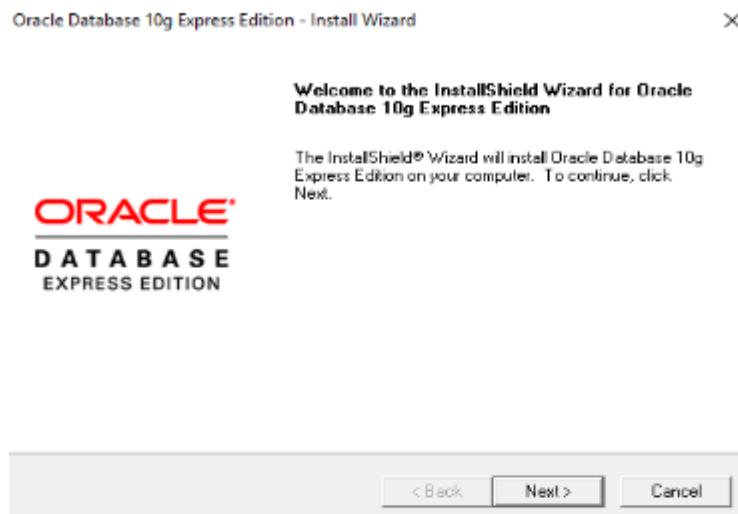
Practical -2

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

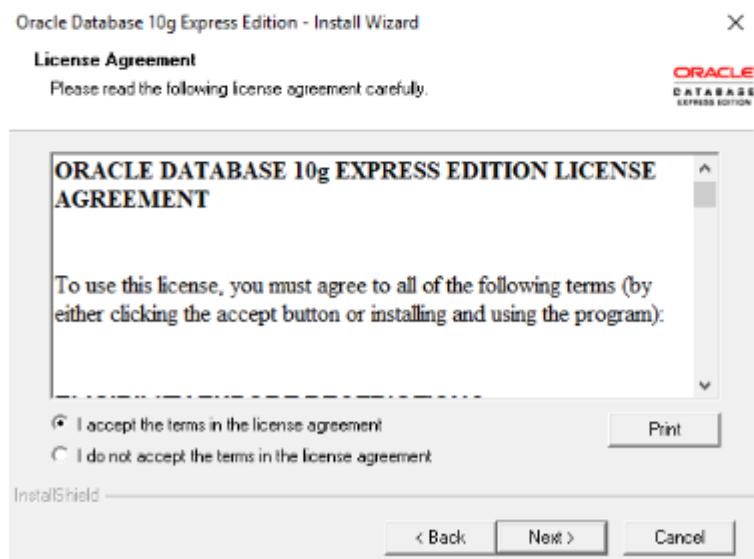
THEORY:



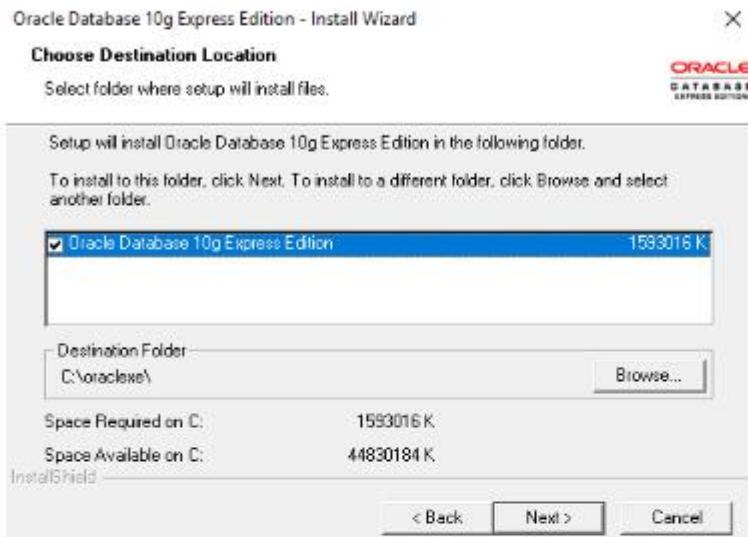
3) Click on Next button



4) Accept license agreement and click on next button



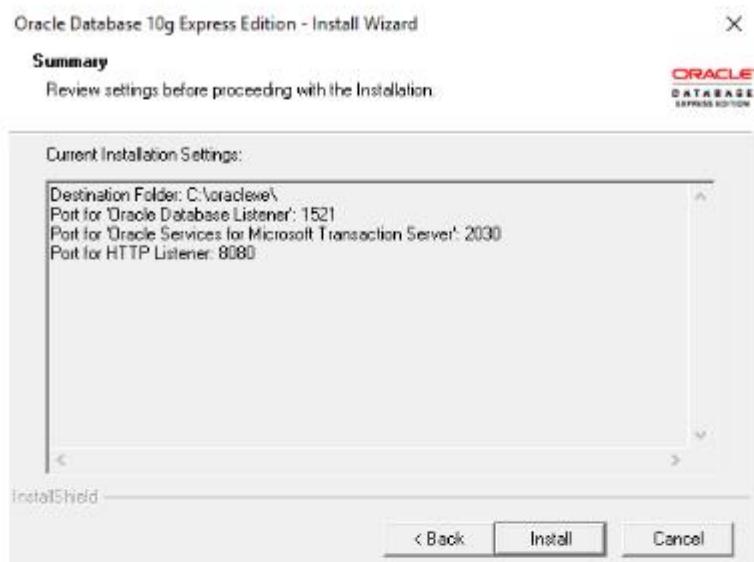
5) Click on next button



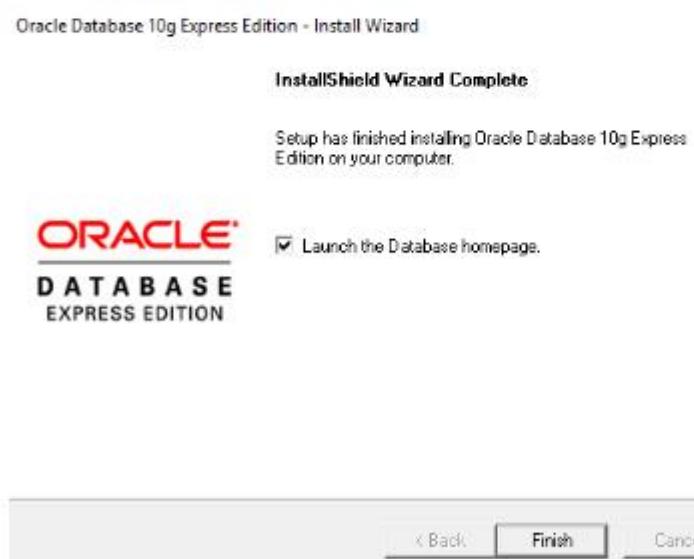
6) Enter password and confirm password for SYS and SYSTEM user. Please remember it because once installation will be over you have to enter it. To make it easy to remember give password as : "oracle"



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)



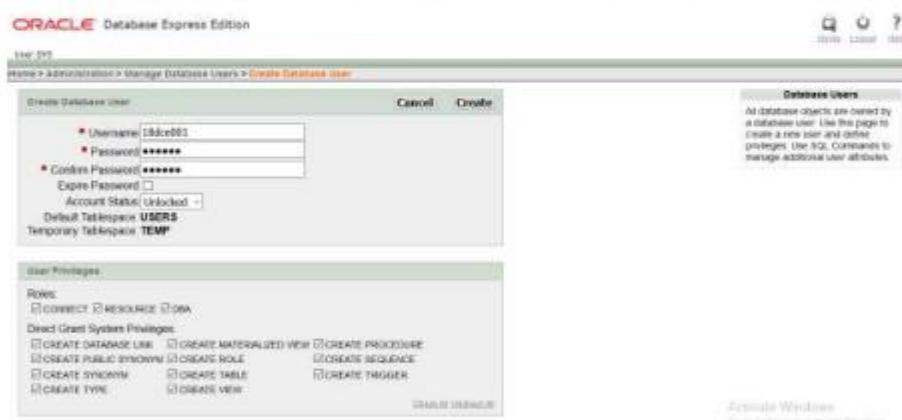
10) Click on Administration



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.

The screenshot shows the Oracle Database Express Edition interface. The title bar reads "ORACLE® Database Express Edition". Below it, a navigation bar shows "User: 19DCE101" and "Home > Administration > Manage Database Users". A success message "User Created." with a green checkmark is displayed. The main area shows a table with two rows. The first row has a user icon, the username "19DCE101", and a status "1". The second row has a user icon, the username "19DCE101", and a status "1". Below the table is a footer with the text "1.3".

14) Click on login

The screenshot shows the Oracle Database Express Edition login page. The title bar reads "ORACLE® Database Express Edition". Below it, a navigation bar shows "User: 19DCE101" and "Home > Administration > Manage Database Users". The main area displays the message "You are now logged out." in bold black text. At the bottom left, there is a "Login" link.

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



16) Click on SQL



17) Click on SQL Commands



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



19) Thank You!!!

Introduction

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 (ie. 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 I learned how to install oracle & learned about sql & pl/sql.

Practical – 3

Aim: (i) To study DDL-CREATE and DML-INSERT Commands.

CREATE:

Syntax:

```
CREATE TABLE <TABLE_NAME> (
    FIELD_NAME <DATA_TYPE>,
);
```

Snapshot:

The screenshot shows a SQL command being run in an Oracle database environment. The command is:

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

The interface includes a toolbar at the top with 'User: 19DCE101', 'Home > SQL > SQL Commands', and a dropdown for 'Autocommit' and 'Display 10'. Below the command is a large empty area for results. At the bottom, there is a navigation bar with tabs for 'Results' (which is selected), 'Explain', 'Describe', 'Saved SQL', and 'History'.

Table created.

0.61 seconds

INSERT:

Syntax:

```
INSERT INTO <TABLE_NAME>
VALUES ('val-1', 'val-2',.... );
```

Snapshot:

User: 19DCE101

Home > SQL > SQL Commands

Autocommit Display 10 ▾

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

Results Explain Describe Saved SQL History

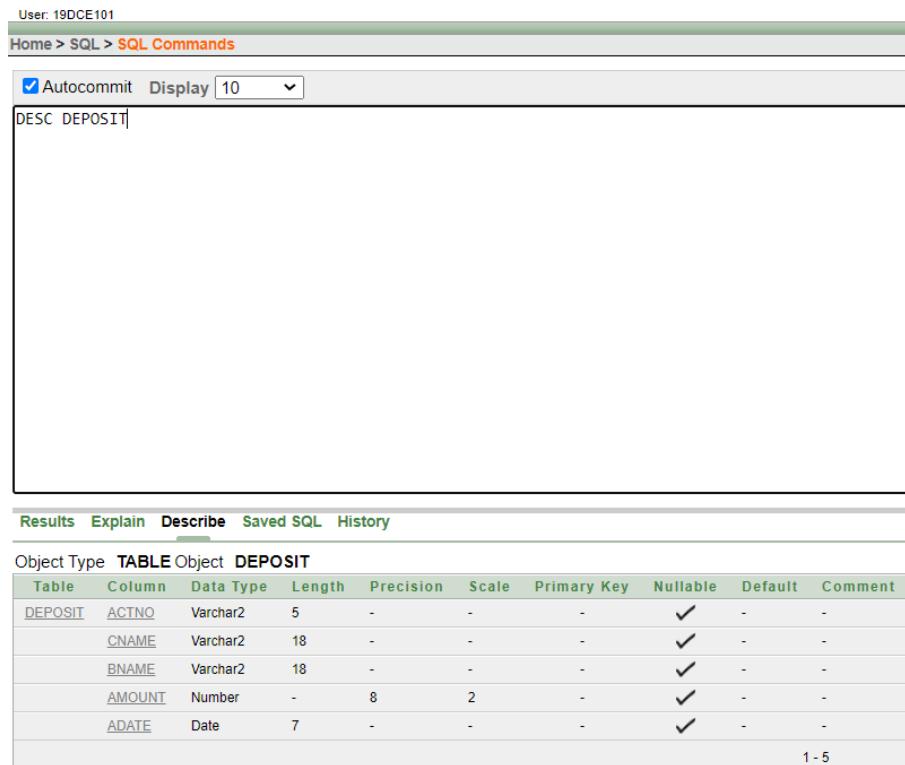
9 row(s) inserted.

0.61 seconds

From the above given tables perform the following queries:

- (1) Describe deposit, branch.

Snapshot:



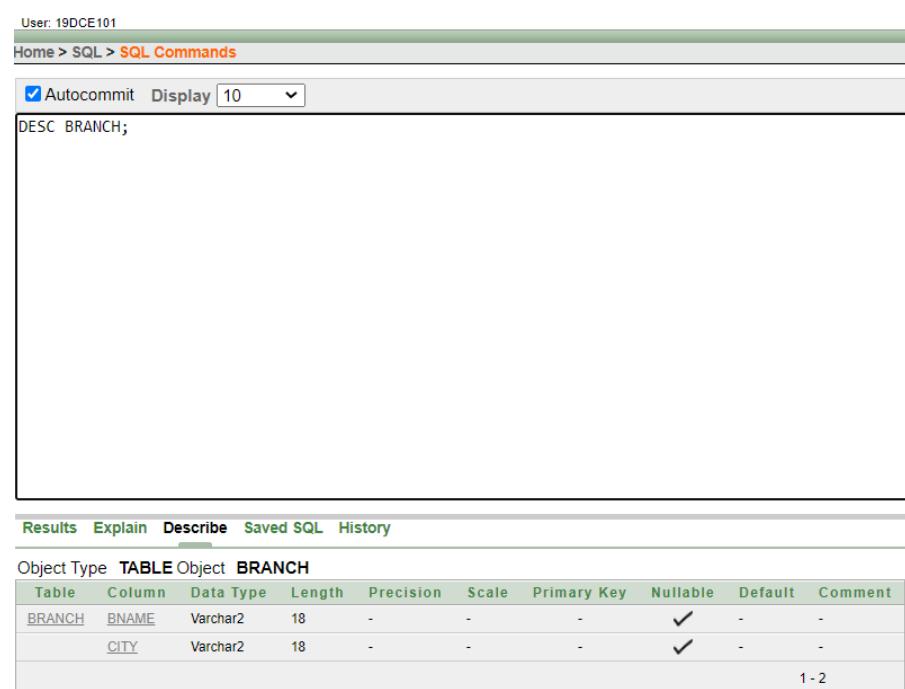
User: 19DCE101
Home > SQL > SQL Commands
 Autocommit Display 10
DESC DEPOSIT

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



User: 19DCE101
Home > SQL > SQL Commands
 Autocommit Display 10
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

(2) Describe borrow, customers.

Snapshot:

User: 19DCE101

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

User: 19DCE101

Home > SQL > SQL Commands

Autocommit Display 10 ▾

```
DESC CUSTOMERS;
```

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 | 19 | - | - | - | ✓ | - | - |
| | CITY | Varchar2 | 18 | - | - | - | ✓ | - | - |

1 - 2

(3) List all data from table DEPOSIT.

Snapshot:

User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10
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 | PRMOD | 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 0.14 seconds [CSV Export](#)

(4) List all data from table BORROW.

Snapshot:

User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10
SELECT * FROM BORROW;

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

6 rows returned in 0.00 seconds [CSV Export](#)

(5) List all data from table CUSTOMERS.

Snapshot:

User: 19DCE101
Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM CUSTOMERS;
```

Results Explain Describe Saved SQL History

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

10 rows returned in 0.01 seconds [CSV Export](#)

(6) List all data from table BRANCH.

Snapshot:

User: 19DCE101
Home > SQL > SQL Commands

Autocommit Display 10

```
SELECT * FROM BRANCH;
```

Results Explain Describe Saved SQL History

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

10 rows returned in 0.00 seconds [CSV Export](#)

(7) Give account no and amount of depositors.

Snapshot:

User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10
SELECT ACTNO, AMOUNT FROM DEPOSIT

| 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.

Snapshot:

User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10
SELECT CNAME FROM DEPOSIT WHERE AMOUNT > 4000

| CNAME |
|--------|
| SUNIL |
| KRANTI |
| MINU |

3 rows returned in 0.08 seconds [CSV Export](#)

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

Snapshot:

User: 19DCE101
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.

Snapshot:

User: 19DCE101
Home > SQL > SQL Commands

Autocommit Display 10 ▾

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

Results Explain Describe Saved SQL History

| CITY |
|-------|
| DELHI |

1 rows returned in 0.02 seconds [CSV Export](#)

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

Snapshot:

User: 19DCE101

Home > SQL > SQL Commands

Autocommit Display 10 ▾

```
SELECT CNAME FROM DEPOSIT WHERE ADATE BETWEEN '1-DEC-96' AND '1-JUN-96'
```

Results Explain Describe Saved SQL History

no data found

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

Snapshot:

User: 19DCE101

Home > SQL > SQL Commands

Autocommit Display 10 ▾

```
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: We Learned how to create tables, insert data into tables & read from the table as any required query.

Practical -4

Perform following queries

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

Snap-Shot:

Query 1: SELECT * FROM DEPOSIT

| A_NO | CNAME | BNAME | AMOUNT | A_DATE |
|------|-------|------------|--------|-----------|
| 101 | ANIL | ANDHERI | 7000 | 01-JAN-06 |
| 102 | SUNIL | VIRAR | 5000 | 15-JUL-06 |
| 103 | JAY | VILLIPARLE | 6500 | 12-MAR-06 |
| 104 | VJAY | 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 [CSV Export](#)

Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

Query 2: SELECT * FROM JOBS

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

6 rows returned in 0.02 seconds [CSV Export](#)

Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

The screenshot shows the Oracle Database Express Edition interface. A SQL command window is open with the URL 127.0.0.1:8080/apex/f?p=4500:1003:159504161478125::NO:::. The query SELECT * FROM EMPLOYEES; is run, and the results are displayed in a table:

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|
| 101 | Smith | 600 | - | 20 | shah | machine learning | it_mgr | Toronto | 105 | 09-AUG-96 |
| 102 | Snehal | 1600 | 300 | 25 | pugta | data science | lec | Las Vegas | - | 14-MAR-96 |
| 103 | Adama | 1100 | 0 | 20 | wales | Machine Learning | mi_mgr | Ontario | 105 | 30-NOV-95 |
| 104 | Aman | 3000 | - | 15 | sharma | Virtual Reality | comp_op | Mexico | 12 | 02-OCT-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 | it_acc | Melbourne | 105 | 28-SEP-97 |
| 107 | Anamika | 2975 | - | 30 | jha | Artificial Intelligence | it_prog | New York | - | 15-JUL-97 |

7 rows returned in 0.00 seconds [CSV Export](#)

Language: en-us Application Express 2.1.0.09.39 Copyright © 1999, 2006, Oracle. All rights reserved.

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. A SQL command window is open with the URL 127.0.0.1:8080/apex/f?p=4500:1003:159504161478125::NO:::. The query SELECT A_NO, AMOUNT FROM DEPOSITZ WHERE A_DATE BETWEEN '01-JAN-06' AND '25-JUN-06'; is run, and the results are displayed in a table:

| A_NO | AMOUNT |
|------|--------|
| 101 | 7000 |
| 103 | 6500 |

2 rows returned in 0.00 seconds [CSV Export](#)

Language: en-us Application Express 2.1.0.09.39 Copyright © 1999, 2006, Oracle. All rights reserved.

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following SQL code:

```
SELECT * FROM JOB WHERE MIN_SAL > 4000
```

The Results tab displays the output of the query:

| JOB_ID | JOB_TITLE | MIN_SAL | MAX_SAL |
|--------|-------------------|---------|---------|
| MK_MGR | MARKETING MANAGER | 9000 | 12000 |
| FL_MGR | FINANCE MANAGER | 6000 | 12000 |
| PL_ACC | ACCOUNT | 4200 | 9000 |
| LEC | LECTURER | 6000 | 17000 |

4 rows returned in 0.00 seconds

Language: en-us Application Express 2.1.0.0.39 Copyright © 1999, 2006, Oracle. All rights reserved.

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following SQL code:

```
SELECT EMP_NAME AS NAME, EMP_SAL FROM EMPLOYEE WHERE DEPT_NO = 20
```

The Results tab displays the output of the query:

| NAME | EMP_SAL |
|-------|---------|
| Smith | 800 |
| Adama | 1100 |

2 rows returned in 0.00 seconds

Language: en-us Application Express 2.1.0.0.39 Copyright © 1999, 2006, Oracle. All rights reserved.

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

Snap-Shot:

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

| EMP_NO | EMP_NAME | DEPT_NAME |
|--------|----------|--------------------|
| 101 | Smith | machine learning |
| 103 | Adama | machine learning |
| 104 | Aman | virtual reality |
| 105 | Anita | big data analytics |
| 106 | Sneha | big data analytics |

5 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

```
SELECT * FROM EMPLOYEE WHERE MANAGER_ID IS NOT NULL AND EMP_COMM IS NOT NULL;
```

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|--------------------|----------|-----------|------------|-----------|
| 103 | Adama | 1100 | 0 | 20 | wales | machine learning | rml_mgr | ontario | 105 | 30-NOV-95 |
| 105 | Anita | 5000 | 50000 | 10 | peter | big data analytics | comp_cpr | germany | 107 | 01-JAN-96 |
| 106 | Sneha | 2450 | 24500 | 10 | joseph | big data analytics | ft_acc | melbourne | 105 | 26-SEP-97 |

3 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

```
SELECT A_NO || ' ' || CNAME AS REQUIRED_TABLE FROM DEPOSIT2 WHERE AMOUNT != 8000
```

| REQUIRED_TABLE |
|----------------|
| 101 ANIL |
| 102 SUNIL |
| 103 JAY |
| 105 KEYUR |
| 106 MAYUR |

5 rows returned in 0.00 seconds [CSV Export](#)

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

Snap-Shot:

```
SELECT * FROM JOBS WHERE MIN_SAL = 2000 OR MIN_SAL = 4000
```

| JOB_ID | JOB_TITLE | MIN_SAL | MAX_SAL |
|---------|------------|---------|---------|
| IT_PROG | PROGRAMMER | 4000 | 10000 |
| CLERK | CLERK | 2000 | 5000 |

1 rows returned in 0.01 seconds [CSV Export](#)

To study various options of LIKE predicate

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following query is entered:

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

The results pane displays the following data:

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|----------|------------|-----------|
| 103 | Adama | 1100 | 0 | 20 | vales | machine learning | ml_mgr | ontario | 105 | 30-NOV-95 |
| 104 | Aman | 3000 | - | 15 | sharma | virtual reality | comp-op | mexico | 12 | 02-OCT-97 |
| 107 | Anamika | 2875 | - | 30 | jha | artificial intelligence | it_prog | new york | - | 15-JUL-97 |

3 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following query is entered:

```
SELECT EMP_NO,EMP_NAME,EMP_SAL FROM EMPLOYEE WHERE EMP_NAME LIKE 'Ani____'
```

The results pane displays the following data:

| EMP_NO | EMP_NAME | EMP_SAL |
|--------|----------|---------|
| 105 | Anta | 5000 |

1 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

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

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|
| 101 | Smith | 800 | - | 20 | shah | machine learning | it_mgr | Toronto | 105 | 09-AUG-99 |
| 102 | Snehal | 1500 | 300 | 25 | gupta | data science | it_c | Las Vegas | - | 14-MAR-99 |
| 104 | Aman | 3000 | - | 15 | sharma | virtual reality | comp_op | Mexico | 12 | 02-OCT-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 | it_acc | Melbourne | 105 | 26-SEP-97 |
| 107 | Anamika | 2975 | - | 30 | jha | artificial intelligence | it_prog | New York | - | 15-JUL-97 |

6 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

```
SELECT CNAME FROM DEPOSIT2 WHERE BNAME = 'ANDHERI' OR BNAME = 'DADAR' OR BNAME = 'VIRAR'
```

| 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_ '.

Snap-Shot:

```
SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO:::
SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE 'FI_%'
```

| JOB_TITLE |
|-----------------|
| FINANCE MANAGER |
| ACCOUNT |

2 rows returned in 0.00 seconds CSV Export

Language en-us Application Express 7.1.0.0.39 Copyright © 1999-2006 Oracle All rights reserved.

(6) Display the title/name of job who's last three character are '_MGR' and their maximum salary is greater than Rs 12000.

Snap-Shot:

```
SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO:::
SELECT JOB_TITLE FROM JOB WHERE JOB_ID LIKE '%_MGR' AND MAX_SAL > 12000
```

| JOB_TITLE |
|-------------------|
| MARKETING MANAGER |

1 rows returned in 0.00 seconds CSV Export

Language en-us Application Express 7.1.0.0.39 Copyright © 1999-2006 Oracle All rights reserved.

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. The query entered is:

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

The results table shows two rows:

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|--------------------|---------|-----------|------------|-----------|
| 105 | Anta | 5000 | 50000 | 10 | patel | big data analytics | comp_op | germany | 107 | 01-JAN-98 |
| 106 | Sneha | 2450 | 24500 | 10 | joseph | big data analytics | it_acc | melbourne | 105 | 26-SEP-97 |

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’.

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. The query entered is:

```
SELECT * FROM EMPLOYEE WHERE EMP_COMM IS NULL AND MANAGER_ID IS NULL AND EMP_NAME LIKE '__a__'
```

The results table shows one row:

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|----------|------------|-----------|
| 107 | Anamika | 2075 | - | 30 | jha | artificial intelligence | it_prog | new york | - | 15-JUL-97 |

1 rows returned in 0.02 seconds CSV Export

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. A SQL query is entered in the command window:

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

The results pane displays the following table:

| JOB_ID | JOB_TITLE | MIN_SAL | MAX_SAL |
|---------|-------------------|---------|---------|
| IT_PROG | PROGRAMMER | 4000 | 10000 |
| MK_MGR | MARKETING MANAGER | 9000 | 15000 |
| FL_MGR | FINANCE MANAGER | 8200 | 12000 |
| FL_ACC | ACCOUNT | 4200 | 9000 |
| COMP_OP | COMPUTER OPERATOR | 1500 | 3000 |

5 rows returned in 0.00 seconds [CSV Export](#)

Language: en-us Application Express 2.1.0.00.39 Copyright © 1999, 2006, Oracle. All rights reserved.

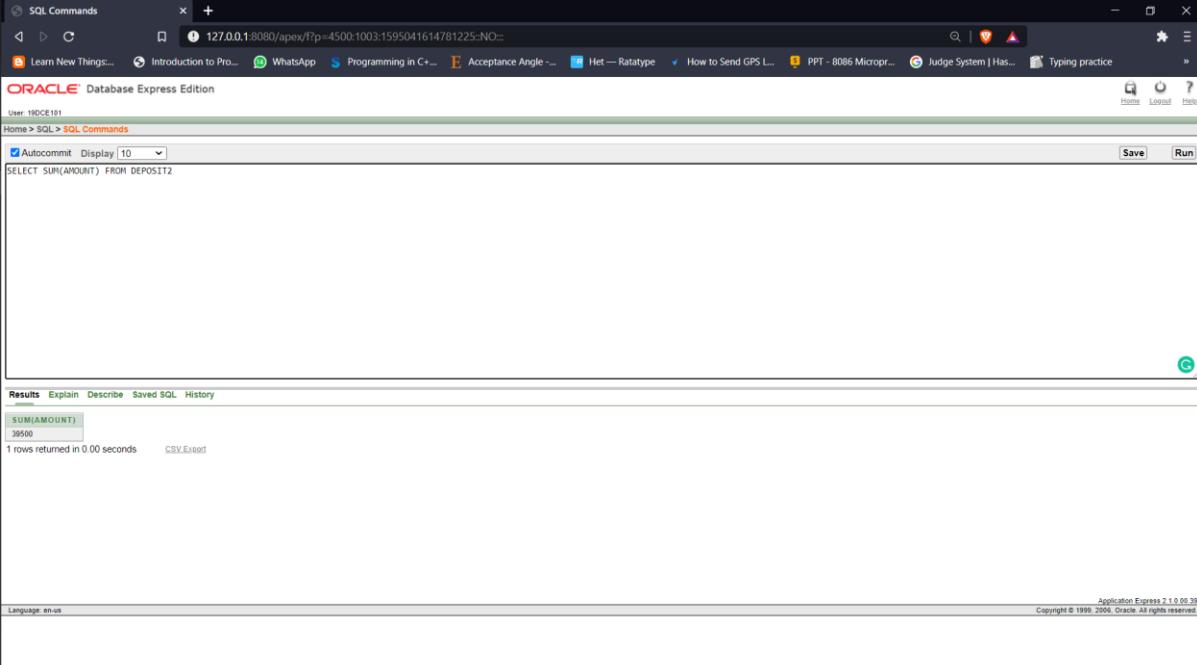
Conclusion: We Learned how to use Like Predicate & perform specific queries.

Practical -5

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

(1) List total deposit from deposit.

Snap-Shot:



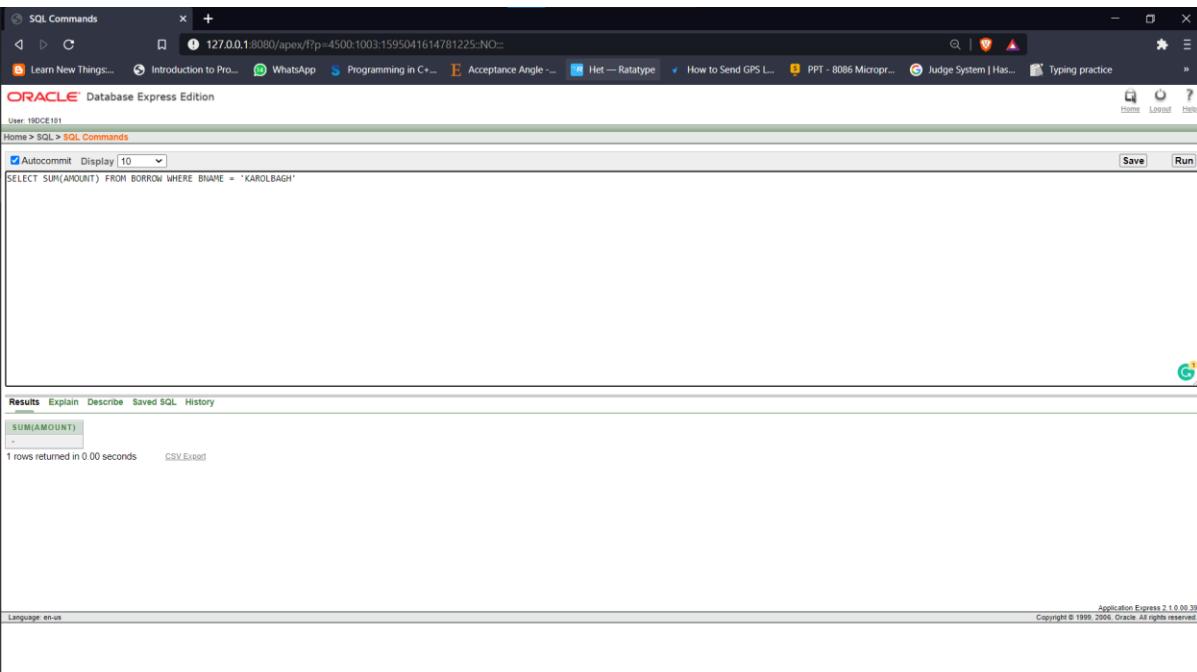
The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL query is executed:

```
SELECT SUM(AMOUNT) FROM DEPOSIT2;
```

The results show a single row with the value 39500. The interface includes tabs for Results, Explain, Describe, Saved SQL, and History. The bottom status bar indicates "Language: en-us".

(2) List total loan from karolbagh branch

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL query is executed:

```
SELECT SUM(AMOUNT) FROM BORROW WHERE BNNAME = 'KAROLBAGH';
```

The results show a single row with the value -. The interface includes tabs for Results, Explain, Describe, Saved SQL, and History. The bottom status bar indicates "Language: en-us".

(3) Give maximum loan from branch vrce.

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL query is entered:

```
SELECT MAX(AMOUNT) FROM BORROW WHERE BRANCH = 'VRCE'
```

The Results tab displays the output:

| MAX(AMOUNT) |
|-------------|
| 1000 |

1 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.00.39 Copyright © 1999, 2006, Oracle. All rights reserved.

(4) Count total number of customers

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL query is entered:

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

The Results tab displays the output:

| COUNT(CNAME) |
|--------------|
| 10 |

1 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.00.39 Copyright © 1999, 2006, Oracle. All rights reserved.

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

Snap-Shot:

The screenshot shows a SQL Commands window in Oracle Database Express Edition. The query entered is:

```
SELECT COUNT(CITY) FROM CUSTOMERS;
```

The results section shows a single row with the value 10 under the column 'COUNT(CITY)'. Below the results, it says '1 rows returned in 0.00 seconds' and has CSV Export and Language en-us options.

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

Snap-Shot:

The screenshot shows a SQL Commands window in Oracle Database Express Edition. The query entered is:

```
CREATE TABLE SUPPLIER1 AS (SELECT * FROM EMPLOYEE)
SELECT * FROM SUPPLIER1
```

The results section shows the data from the EMPLOYEE table, which includes columns like EMP_NO, EMP_NAME, EMP_SAL, etc. Below the results, it says '7 rows returned in 0.01 seconds' and has CSV Export and Language en-us options.

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL code is run:

```
CREATE TABLE SUP_1 AS (SELECT EMP_NO,EMP_NAME FROM EMPLOYEE)
SELECT * FROM SUP_1
```

The Results tab displays the data returned by the query:

| EMP_NO | EMP_NAME |
|--------|----------|
| 101 | Smith |
| 102 | Singh |
| 103 | Adama |
| 104 | Aman |
| 105 | Anita |
| 106 | Sneha |
| 107 | Anamika |

7 rows returned in 0.04 seconds

(8) Create table sup2 from employee with no data

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL code is run:

```
CREATE TABLE SUP_2 AS (SELECT * FROM EMPLOYEE WHERE EMP_NO = 2000)
SELECT * FROM SUP_2
```

The Results tab displays the message:

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.

Snap-Shot:

```
SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:159504161478125::NO::
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit: Display: 10
INSERT INTO SUP_2 (SELECT * FROM EMPLOYEE WHERE EMP_NAME LIKE '_n____')
SELECT * FROM SUP_2

Results Explain Describe Saved SQL History
EMP_NO EMP_NAME EMP_SAL EMP_COMM DEPT_NO L_NAME DEPT_NAME JOB_ID LOCATION MANAGER_ID HIREDATE
105 Anta 5000 50000 10 patel big data analytics comp_op germany 107 01-JAN-98
106 Sneha 2450 24500 10 joseph big data analytics fl_acc melbourne 105 26-SEP-97
2 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.00.39
Copyright © 1999, 2006, Oracle. All rights reserved.
```

(10) Delete all the rows from sup1.

Snap-Shot:

```
SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:159504161478125::NO::
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit: Display: 10
DELETE FROM SUP_1

Results Explain Describe Saved SQL History
7 row(s) deleted.
0.00 seconds

Language: en-us Application Express 2.1.0.00.39
Copyright © 1999, 2006, Oracle. All rights reserved.
```

(11) Delete the detail of supplier whose sup_no is 103.

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. In the SQL editor, the command `DELETE FROM SUPPLIER1 WHERE EMP_NO = 103` is entered. After execution, the results show "1 row(s) deleted." and "0.00 seconds". The status bar at the bottom right indicates "Application Express 2.1.0.0.39" and "Copyright © 1999, 2006, Oracle. All rights reserved."

The screenshot shows the Oracle Database Express Edition SQL Commands interface. In the SQL editor, the command `DELETE FROM SUPPLIER1 WHERE EMP_NO = 103` is entered. After execution, the results show "1 row(s) deleted." and "0.00 seconds". Below this, a `SELECT * FROM SUPPLIER1` query is run, and the results table displays 6 rows of data. The status bar at the bottom right indicates "Application Express 2.1.0.0.39" and "Copyright © 1999, 2006, Oracle. All rights reserved."

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|
| 101 | Smith | 600 | - | 20 | shah | machine learning | it_mngr | Toronto | 105 | 06-AUG-96 |
| 102 | Snehal | 1600 | 300 | 25 | pulta | data science | it_lec | las vegas | - | 14-MAR-95 |
| 104 | Aman | 3000 | - | 15 | sharma | virtual reality | comp_op | mexico | 12 | 02-OCT-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 | it_acc | melbourne | 105 | 26-SEP-97 |
| 107 | Anamika | 2875 | - | 30 | jha | artificial intelligence | it_prog | new york | - | 15-JUL-97 |

(12) Rename the table sup2.

Snap-Shot:

```
ALTER TABLE SUP_2 RENAME TO SUPPLIER2
SELECT * FROM SUPPLIER2
```

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | 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-96 |
| 106 | Shelia | 2450 | 24500 | 10 | joseph | big data analytics | ft_acc | melbourne | 105 | 26-SEP-97 |

2 rows returned in 0.00 seconds [CSV Export](#)

(13) Destroy table sup1 with all the data.

Snap-Shot:

```
DROP TABLE SUP_1
SELECT * FROM SUP_1
```

✖ ORA-00942: table or view does not exist

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL code is entered:

```
UPDATE EMPLOYEE SET DEPT_NO = 10 WHERE EMP_NAME LIKE '_m%';
SELECT * FROM EMPLOYEE;
```

The results section displays the updated employee data:

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|
| 101 | Smith | 600 | - | 10 | shah | machine learning | it_mngr | Toronto | 105 | 09-AUG-96 |
| 102 | Snehal | 1600 | 300 | 25 | gupta | data science | itc | Las Vegas | - | 14-MAR-96 |
| 103 | Adama | 1100 | 0 | 20 | wales | machine learning | mi_mngr | ontario | 105 | 30-NOV-95 |
| 104 | Aman | 3000 | - | 10 | sharma | virtual reality | comp_op | mexico | 12 | 02-OCT-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 | it_acc | melbourne | 105 | 26-SEP-97 |
| 107 | Anamika | 2975 | - | 30 | jha | artificial intelligence | it_prog | New York | - | 15-JUL-97 |

7 rows returned in 0.00 seconds

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL code is entered:

```
UPDATE EMPLOYEE SET EMP_NAME = 'RAMESH' WHERE EMP_NO = 103;
SELECT * FROM EMPLOYEE;
```

The results section displays the updated employee data:

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|
| 101 | Smith | 600 | - | 10 | shah | machine learning | it_mngr | Toronto | 105 | 09-AUG-96 |
| 102 | Snehal | 1600 | 300 | 25 | gupta | data science | itc | Las Vegas | - | 14-MAR-96 |
| 103 | RAMESH | 1100 | 0 | 20 | wales | machine learning | mi_mngr | ontario | 105 | 30-NOV-95 |
| 104 | Aman | 3000 | - | 10 | sharma | virtual reality | comp_op | mexico | 12 | 02-OCT-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 | it_acc | melbourne | 105 | 26-SEP-97 |
| 107 | Anamika | 2975 | - | 30 | jha | artificial intelligence | it_prog | New York | - | 15-JUL-97 |

7 rows returned in 0.00 seconds

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands tab, the following SQL code is entered:

```
ALTER TABLE EMPLOYEE ADD PHONE NUMBER(10)
SELECT * FROM EMPLOYEE;
```

The results section displays the current data from the EMPLOYEE table, which includes columns: EMP_NO, EMP_NAME, EMP_SAL, EMP_COMM, DEPT_NO, L_NAME, DEPT_NAME, JOB_ID, LOCATION, MANAGER_ID, and HIREDATE. A new column, PHONE, is added at the end of the table structure. The results table shows 10 rows of data. At the bottom, it says "7 rows returned in 0.01 seconds".

(17) Modify the column emp_name to hold maximum of 30 characters.

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands tab, the following SQL code is entered:

```
ALTER TABLE EMPLOYEE MODIFY EMP_NAME VARCHAR(30);
```

The results section shows the message "Table altered." and "0.25 seconds".

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

Snap-Shot:

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

The screenshot shows the Oracle Database Express Edition interface. The SQL command `SELECT COUNT(DISTINCT DEPT_NO) FROM EMPLOYEE WHERE EMP_SAL > 1000;` is entered in the SQL Commands window. The results show a single row with the value 4, indicating there are 4 distinct department numbers for employees with a salary greater than 1000.

| COUNT(DISTINCT DEPT_NO) |
|-------------------------|
| 4 |

1 rows returned in 0.01 seconds

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

Snap-Shot:

```
SELECT * FROM EMPLOYEE ORDER BY EMP_NAME,EMP_NO DESC;
```

The screenshot shows the Oracle Database Express Edition interface. The SQL command `SELECT * FROM EMPLOYEE ORDER BY EMP_NAME,EMP_NO DESC;` is entered in the SQL Commands window. The results display all employee details, ordered first by employee name in ascending order and then by employee number in descending order.

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE | PHONE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|-------|
| 104 | Aman | 3000 | - | 10 | sharma | virtual reality | comp-op | mexico | 12 | 02-OCT-97 | - |
| 107 | Anamika | 2975 | - | 30 | jha | artificial intelligence | it_prog | new york | - | 15-JUL-97 | - |
| 105 | Anta | 50000 | 10 | 10 | patel | big data analytics | comp_op | germany | 107 | 01-JAN-98 | - |
| 103 | RAMESH | 1100 | 0 | 20 | vales | machine learning | ml_mgr | ontario | 105 | 30-NOV-95 | - |
| 101 | Smith | 800 | - | 10 | shah | machine learning | it_mgr | Toronto | 105 | 09-AUG-96 | - |
| 106 | Sneha | 2450 | 24500 | 10 | joseph | big data analytics | it_acc | Melbourne | 105 | 25-SEP-97 | - |
| 102 | Shnehal | 1600 | 300 | 25 | gupta | data science | iec | las vegas | - | 14-MAR-96 | - |

7 rows returned in 0.00 seconds

(20) Display the dept_no in ascending order and accordingly display emp_comm in descending order.

Snap-Shot:

```
SELECT DEPT_NO,EMP_COMM FROM EMPLOYEE ORDER BY DEPT_NO,EMP_COMM DESC
```

| DEPT_NO | EMP_COMM |
|---------|----------|
| 10 | - |
| 10 | - |
| 10 | 50000 |
| 10 | 24500 |
| 20 | 0 |
| 25 | 300 |
| 30 | - |

7 rows returned in 0.00 seconds CSV Export

(21) Update the value of emp_comm to 500 where dept_no is 20.

Snap-Shot:

```
UPDATE EMPLOYEE SET EMP_COMM = 500 WHERE DEPT_NO = 20
SELECT * FROM EMPLOYEE
```

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE | PHONE |
|--------|----------|---------|----------|---------|---------|-------------------------|---------|-----------|------------|-----------|-------|
| 101 | Smith | 800 | - | 10 | shah | machine learning | it_mgr | Toronto | 105 | 09-AUG-99 | - |
| 102 | Snehal | 1500 | 300 | 25 | p Gupta | data science | itc | Las Vegas | - | 14-MAR-99 | - |
| 103 | RAMESH | 1100 | 500 | 20 | valma | Machine Learning | ml_mgr | Ontario | 105 | 30-NOV-95 | - |
| 104 | Aman | 3000 | - | 10 | Sharma | Virtual Reality | comp_op | Mexico | 12 | 02-OCT-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 | it_acc | Melbourne | 105 | 26-SEP-97 | - |
| 107 | Anamika | 2075 | - | 30 | Jha | Artificial Intelligence | it_prog | New York | - | 15-JUL-97 | - |

7 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

```
SELECT * FROM EMPLOYEE ORDER BY emp_comm NULLS FIRST, emp_no DESC
```

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE | PHONE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|-------|
| 102 | Snehal | 1600 | 300 | 25 | gupta | data science | iec | las vegas | - | 14-NOV-95 | - |
| 103 | RAMESH | 1100 | 500 | 20 | wales | machine learning | ml_mgr | ontario | 105 | 30-NOV-95 | - |
| 106 | Sneha | 2450 | 24500 | 10 | joseph | big data analytics | it_acc | melbourne | 105 | 26-SEP-97 | - |
| 105 | Anta | 5000 | 50000 | 10 | patel | big data analytics | comp_op | germany | 107 | 01-JAN-98 | - |
| 104 | Aman | 3000 | - | 10 | sharma | virtual reality | comp_op | mexico | 12 | 02-OCT-97 | - |
| 107 | Anamika | 2975 | - | 30 | jha | artificial intelligence | it_prog | new york | - | 15-JUL-97 | - |
| 101 | Smith | 800 | - | 10 | shin | machine learning | it_mgr | Toronto | 105 | 09-AUG-98 | - |
| 102 | Snehal | 1600 | 300 | 25 | pusta | data science | iec | las vegas | - | 14-MAR-98 | - |
| 103 | RAMESH | 1100 | 500 | 20 | wales | machine learning | ml_mgr | ontario | 105 | 30-NOV-95 | - |
| 106 | Sneha | 2450 | 24500 | 10 | joseph | big data analytics | it_acc | melbourne | 105 | 26-SEP-97 | - |
| 105 | Anta | 5000 | 50000 | 10 | patel | big data analytics | comp_op | germany | 107 | 01-JAN-98 | - |

7 rows returned in 0.00 seconds [CSV Export](#)

Language: en-us Application Express 7.1.0.0.39 Copyright © 1999-2006 Oracle. All rights reserved.

(23) Display the emp_comm in ascending order with null value last and accordingly sort emp_no in descending order.

Snap-Shot:

```
SELECT * FROM EMPLOYEE ORDER BY emp_comm NULLS LAST, emp_no DESC
```

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE | PHONE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|-----------|------------|-----------|-------|
| 102 | Snehal | 1600 | 300 | 25 | gupta | data science | iec | las vegas | - | 14-NOV-95 | - |
| 103 | RAMESH | 1100 | 500 | 20 | wales | machine learning | ml_mgr | ontario | 105 | 30-NOV-95 | - |
| 106 | Sneha | 2450 | 24500 | 10 | joseph | big data analytics | it_acc | melbourne | 105 | 26-SEP-97 | - |
| 105 | Anta | 5000 | 50000 | 10 | patel | big data analytics | comp_op | germany | 107 | 01-JAN-98 | - |
| 107 | Anamika | 2975 | - | 30 | jha | artificial intelligence | it_prog | new york | - | 15-JUL-97 | - |
| 104 | Aman | 3000 | - | 10 | sharma | virtual reality | comp_op | mexico | 12 | 02-OCT-97 | - |
| 101 | Smith | 800 | - | 10 | shin | machine learning | it_mgr | Toronto | 105 | 09-AUG-98 | - |

7 rows returned in 0.00 seconds [CSV Export](#)

Language: en-us Application Express 7.1.0.0.39 Copyright © 1999-2006 Oracle. All rights reserved.

Conclusion: We Learned how to use aggregate functions, DML queries & Sorting.

Practical -6

To study Single-row functions.

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

Snap-Shot:

```
SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO:::
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle ... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
AutoCommit Display 10 Save Run
SELECT CURRENT_DATE "DATE" FROM DUAL

Results Explain Describe Saved SQL History
DATE
16-FEB-21
1 rows returned in 0.00 seconds CSV Export
Language: en-us
Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.
```

(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.

Snap-Shot:

```
SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO:::
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle ... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
AutoCommit Display 10 Save Run
ALTER TABLE EMPLOYEE ADD NEW_SALARY NUMBER(8,2)
UPDATE EMPLOYEE SET NEW_SALARY = EMP_SAL*1.15
SELECT * 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 2818
107 2975 3421
7 rows returned in 0.00 seconds CSV Export
Language: en-us
Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.
```

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

Snap-Shot:

```

ALTER TABLE EMPLOYEE ADD INCREASE NUMBER;
UPDATE EMPLOYEE SET INCREASE = NEW_SALARY - EMP_SAL;
SELECT EMP_NO,EMP_SAL,ROUND(NEW_SALARY,0) "NEW_SALARY", INCREASE FROM EMPLOYEE;

```

| 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 | 2818 | 367.5 |
| 107 | 2975 | 3421 | 446.25 |

(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.

Snap-Shot:

```

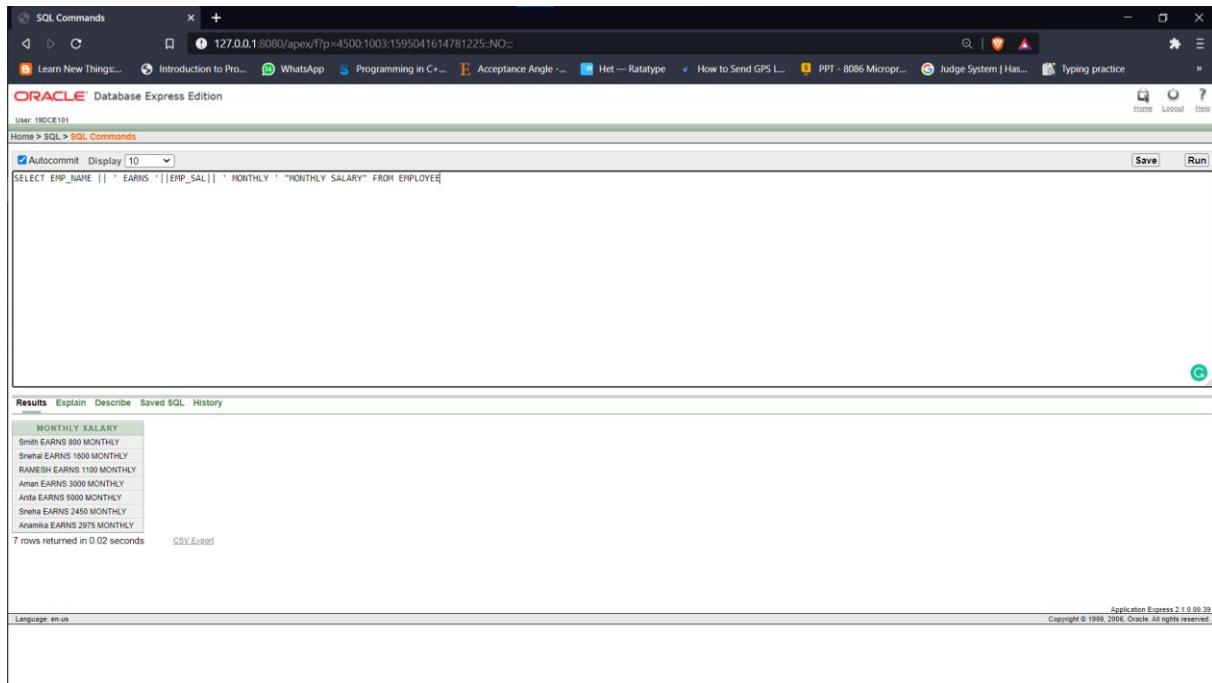
SELECT INITCAP(EMP_NAME) AS "EMPLOYEE NAME", LENGTH(EMP_NAME) AS "LENGTH" FROM EMPLOYEE WHERE EMP_NAME LIKE 'J%' OR EMP_NAME LIKE 'A%' OR EMP_NAME LIKE 'M%' ORDER BY L_NAME DESC;

```

| EMPLOYEE NAME | LENGTH |
|---------------|--------|
| Aman | 4 |
| Anita | 5 |
| Anamika | 7 |

(5) Write a query that produces the following for each employee:
 <employee last name> earns <salary> monthly

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. A SQL command is entered in the SQL Commands window:

```
SELECT EMP_NAME || ' EARNS ' || EMP_SAL || ' MONTHLY ' "MONTHLY SALARY" FROM EMPLOYEE
```

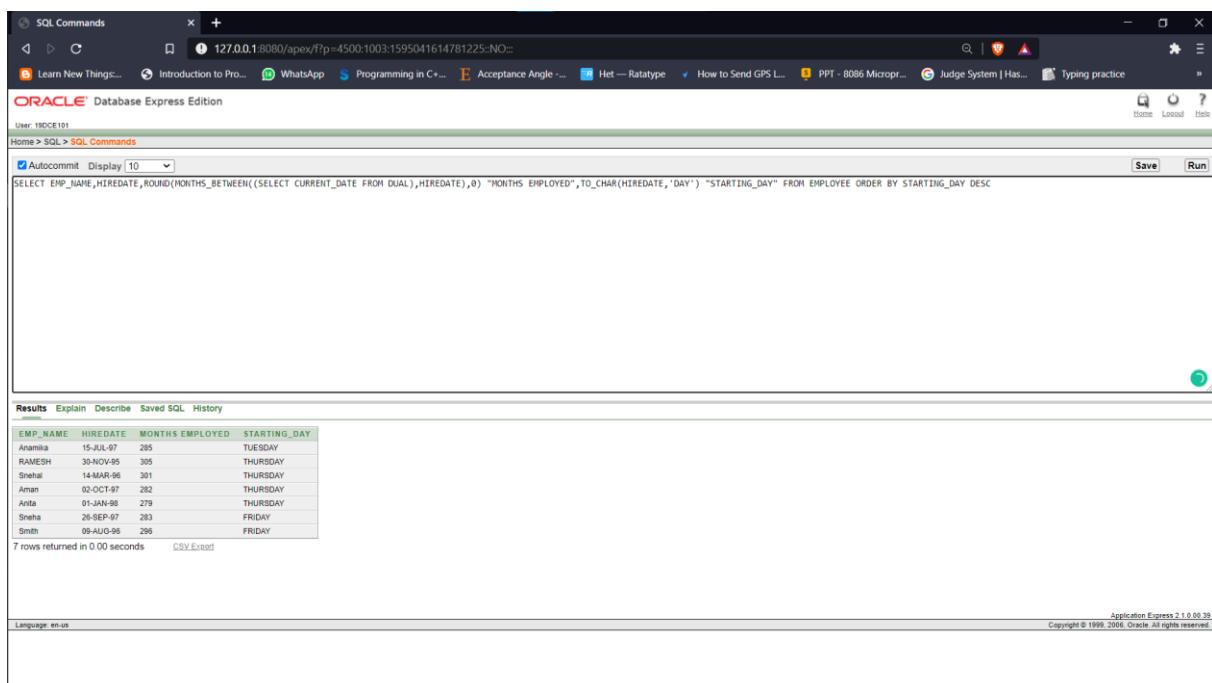
The results are displayed in the Results tab:

| MONTHLY SALARY |
|----------------------------|
| Smith EARNS 800 MONTHLY |
| Sheila EARNS 1600 MONTHLY |
| RAMESH EARNS 1100 MONTHLY |
| Aman EARNS 3000 MONTHLY |
| Anita EARNS 5000 MONTHLY |
| Sheila EARNS 2450 MONTHLY |
| Anamika EARNS 2975 MONTHLY |

7 rows returned in 0.02 seconds

(6) Display the name, 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.

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. A SQL command is entered in the SQL Commands window:

```
SELECT EMP_NAME, HIREDATE, ROUND(MONTHS_BETWEEN((SELECT CURRENT_DATE FROM DUAL),HIREDATE),0) "MONTHS EMPLOYED", TO_CHAR(HIREDATE,'DAY') "STARTING_DAY" FROM EMPLOYEE ORDER BY STARTING_DAY DESC
```

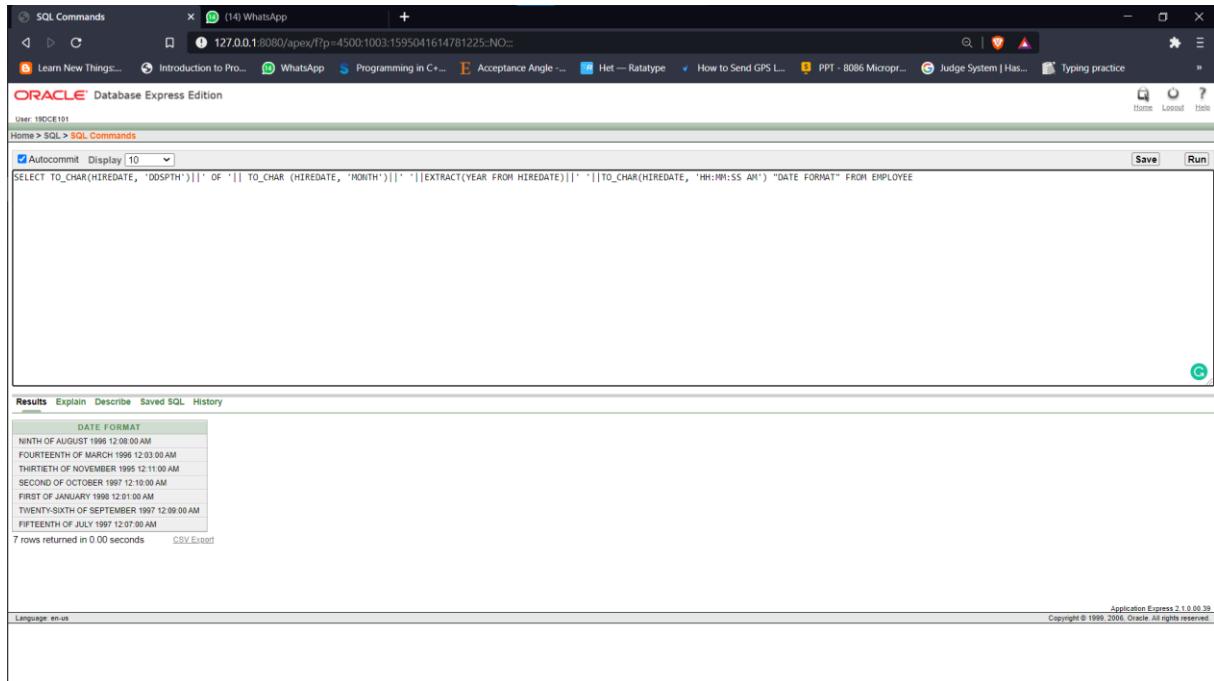
The results are displayed in the Results tab:

| EMP_NAME | HIREDATE | MONTHS EMPLOYED | STARTING_DAY |
|----------|-----------|-----------------|--------------|
| Anamika | 15-JUL-97 | 291 | TUESDAY |
| RAMESH | 30-NOV-95 | 395 | THURSDAY |
| Sheila | 14-MAR-96 | 391 | THURSDAY |
| Aman | 02-OCT-97 | 282 | THURSDAY |
| Anita | 01-JAN-98 | 279 | THURSDAY |
| Sheila | 26-SEP-97 | 283 | FRIDAY |
| Smith | 09-AUG-96 | 299 | FRIDAY |

7 rows returned in 0.00 seconds

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. A SQL command is entered in the SQL Commands window:

```
SELECT TO_CHAR(HIREDATE, 'DOSPTH')||' OF '|| TO_CHAR(HIREDATE, 'MONTH')||' '||EXTRACT(YEAR FROM HIREDATE)||' '||TO_CHAR(HIREDATE, 'HH:MM:SS AM') "DATE FORMAT" FROM EMPLOYEE
```

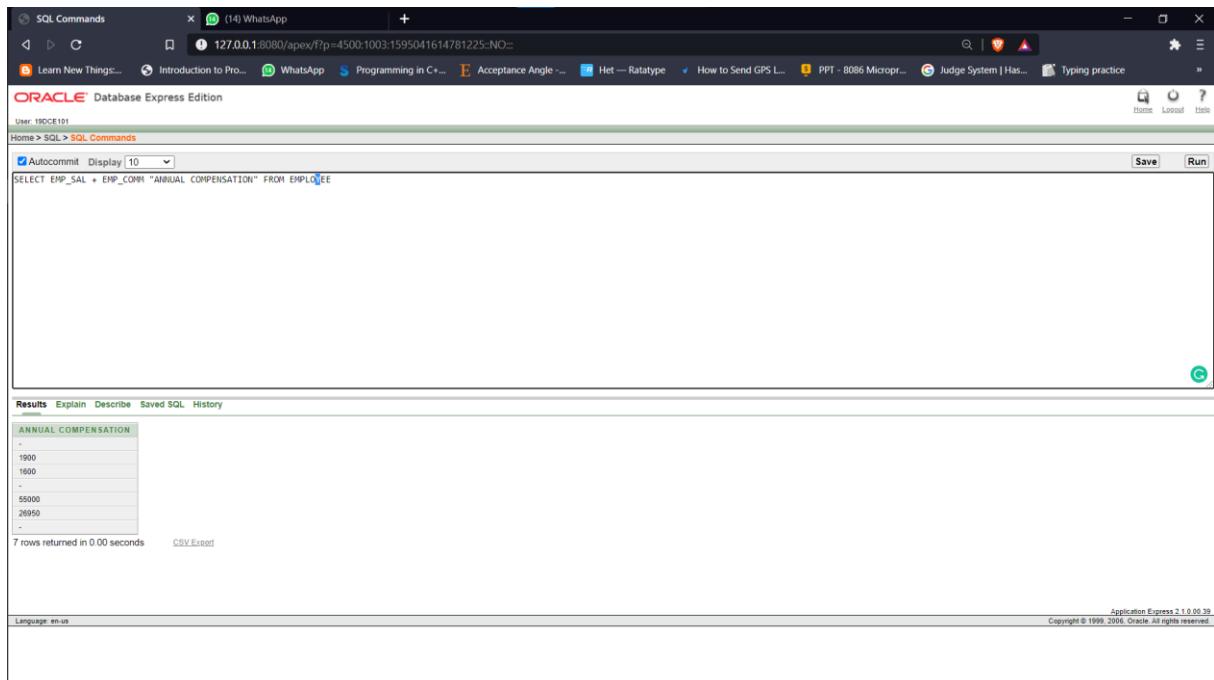
The results window displays the output:

| DATE FORMAT |
|--|
| NINTH OF AUGUST 1996 12:08:00 AM |
| FOURTEENTH OF MARCH 1996 12:03:00 AM |
| THIRTIETH OF NOVEMBER 1995 12:11:00 AM |
| SECOND OF OCTOBER 1997 12:10:00 AM |
| FIRST OF JANUARY 1996 12:01:00 AM |
| TWENTY-SIXTH OF SEPTEMBER 1997 12:09:00 AM |
| FIFTEENTH OF JULY 1997 12:07:00 AM |

7 rows returned in 0.00 seconds

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. A SQL command is entered in the SQL Commands window:

```
SELECT EMP_SAL + EMP_COMM "ANNUAL COMPENSATION" FROM EMPLOYEE
```

The results window displays the output:

| ANNUAL COMPENSATION |
|---------------------|
| - |
| 1800 |
| 1800 |
| - |
| 55000 |
| 26950 |
| - |

7 rows returned in 0.00 seconds

Conclusion: We Learned how to use Single-row functions.

Practical -7

Displaying data from Multiple Tables (join)

(1) Give details of customers ANIL.

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. The SQL command entered is:

```
SELECT DEPOSIT.ACCTNO,CUSTOMERS.CNAME,DEPOSIT.BNAME,DEPOSIT.AMOUNT,DEPOSIT.ADATE FROM CUSTOMERS FULL OUTER JOIN DEPOSIT ON CUSTOMERS.CNAME = DEPOSIT.CNAME WHERE CUSTOMERS.CNAME = 'ANIL';
```

The results pane displays the following data:

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

1 rows returned in 0.21 seconds

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. The SQL command entered is:

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

The results pane displays the following data:

| CNAME |
|----------|
| PRAMOD |
| MANDHURI |

2 rows returned in 0.03 seconds

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

Snap-Shot:

```
SELECT DEPOSIT.CNAME,BRANCH.CITY,CUSTOMERS.CITY FROM DEPOSIT JOIN BRANCH ON DEPOSIT.BNAME = BRANCH.BNAME JOIN CUSTOMERS ON DEPOSIT.CNAME = CUSTOMERS.CNAME WHERE BRANCH.CITY = CUSTOMERS.CITY
```

| CNAME | CITY | CITY |
|---------|--------|--------|
| SHIVANI | BOMBAY | BOMBAY |

1 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

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

| L_NAME | DEPT_NO | DEPT_NAME |
|--------|---------|-------------------------|
| shah | 10 | machine learning |
| gupta | 25 | data science |
| wales | 20 | machine learning |
| sharma | 10 | virtual reality |
| patel | 10 | big data analytics |
| joseph | 10 | big data analytics |
| jha | 30 | artificial intelligence |

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

Snap-Shot:

```
SELECT JOB.JOB_TITLE, EMPLOYEE.LOCATION FROM EMPLOYEE INNER JOIN JOB ON EMPLOYEE.JOB_ID = JOB.JOB_ID WHERE EMPLOYEE.DEPT_NO = 30
```

| JOB_TITLE | LOCATION |
|------------|----------|
| Programmer | new york |

1 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 NEW YORK.

Snap-Shot:

```
SELECT EMP_NAME, DEPT_NO, DEPT_NAME FROM EMPLOYEE WHERE LOCATION = 'new york'
```

| EMP_NAME | DEPT_NO | DEPT_NAME |
|----------|---------|-------------------------|
| Anamika | 30 | artificial intelligence |

1 rows returned in 0.00 seconds [CSV Export](#)

(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.

Snap-Shot:

```
SELECT EMPLOYEE "EMPLOYEE", Emp# "Emp#", MANAGER_ID "Mgr#" FROM EMPLOYEE
```

| EMPLOYEE | Emp# | Mgr# |
|----------|------|------|
| Smith | 101 | 105 |
| Shetal | 102 | - |
| RAMESH | 103 | 105 |
| Aman | 104 | 12 |
| Anita | 105 | 107 |
| Sheila | 106 | 105 |
| Anamika | 107 | - |

7 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

```
SELECT EMPLOYEE, HIREDATE FROM EMPLOYEE WHERE HIREDATE >(SELECT HIREDATE FROM EMPLOYEE WHERE EMP_NO = 101)
```

| EMPLOYEE | HIREDATE |
|----------|-----------|
| Aman | 02-OCT-97 |
| Anita | 01-JAN-98 |
| Sheila | 26-SEP-97 |
| Anamika | 15-JUL-97 |

4 rows returned in 0.00 seconds CSV Export

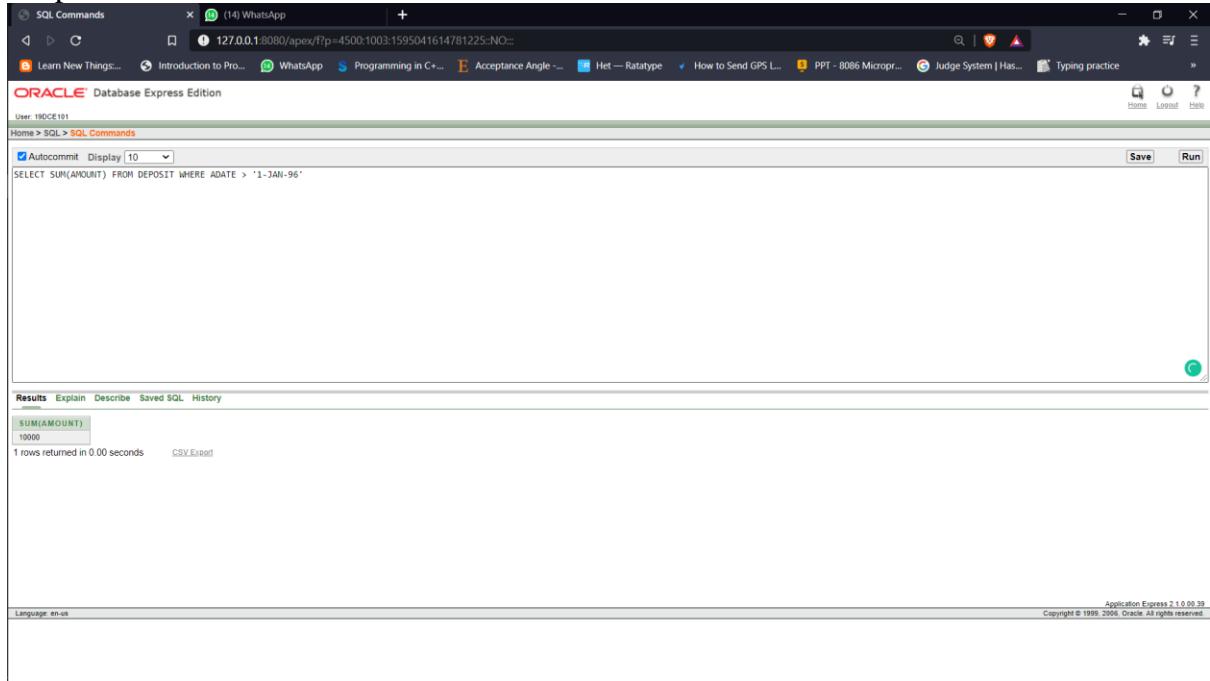
Conclusion: We Learned how to display data using join from multiple tables.

Practical -8

To apply the concept of Aggregating Data using Group functions.

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following SQL query:

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

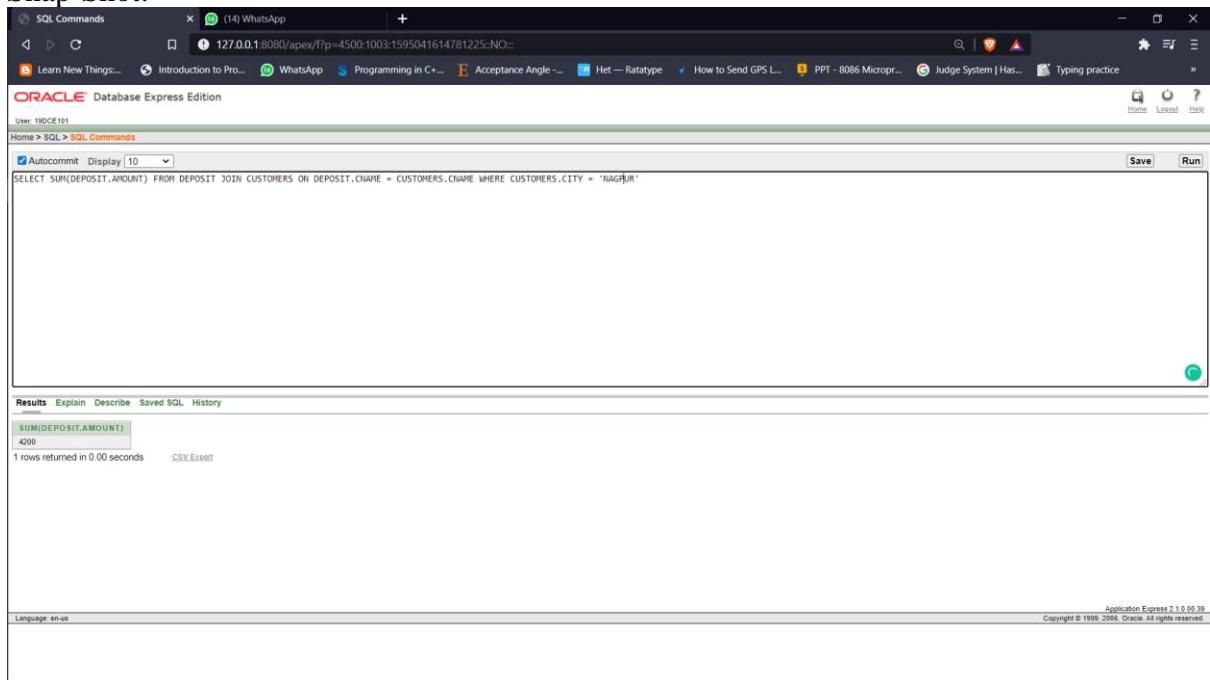
The results section shows the output:

| SUM(AMOUNT) |
|-------------|
| 10000 |

1 rows returned in 0.00 seconds

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following SQL query:

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

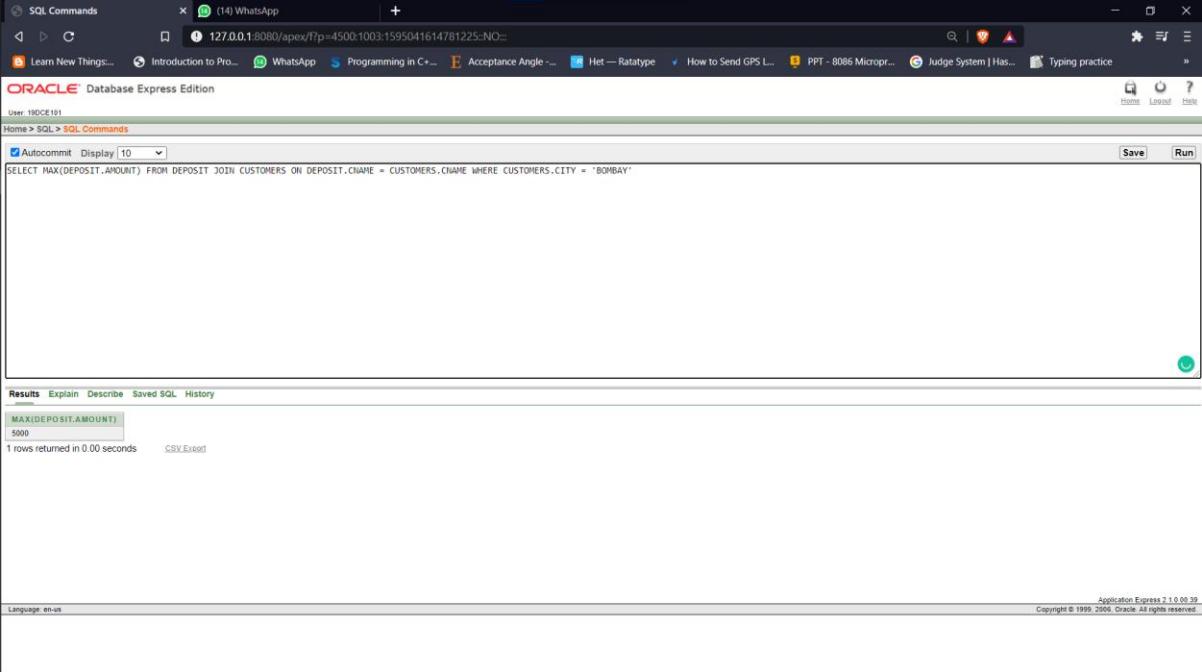
The results section shows the output:

| SUM(DEPOSIT.AMOUNT) |
|---------------------|
| 4200 |

1 rows returned in 0.00 seconds

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL query is entered:

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

The results pane displays the output:

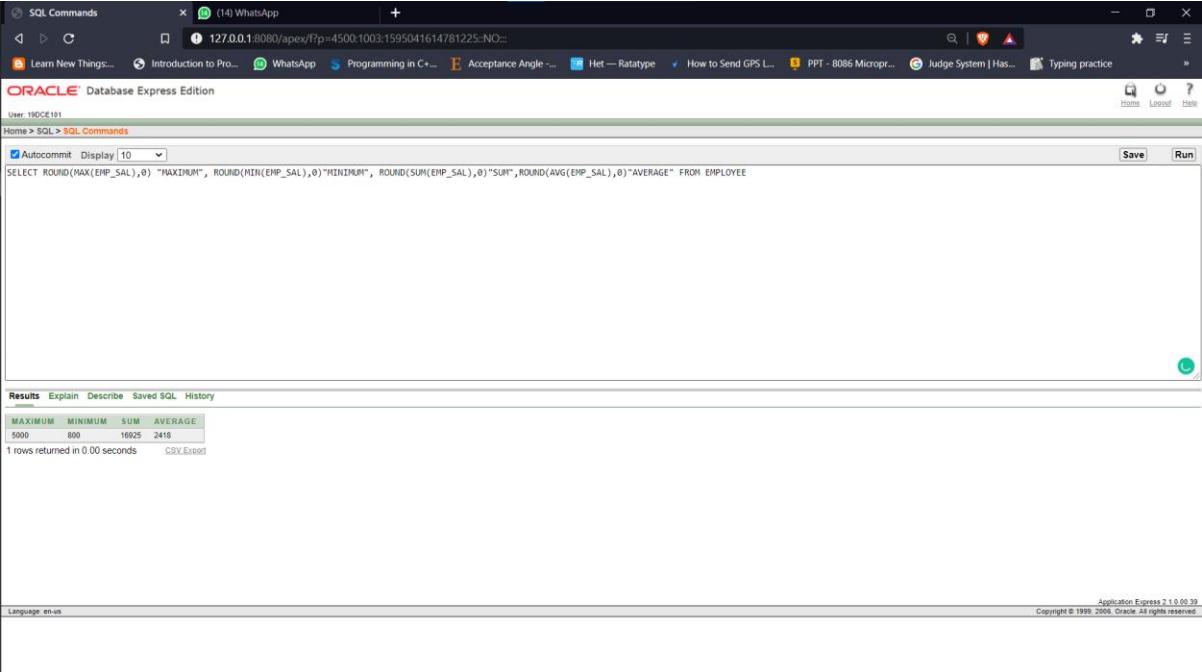
| MAX(DEPOSIT.AMOUNT) |
|---------------------|
| 5000 |

1 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.0.39 Copyright © 1999-2006, Oracle. All rights reserved.

(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.

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL query is entered:

```
SELECT ROUND(MAX(EMP_SAL),0) "MAXIMUM", ROUND(MIN(EMP_SAL),0)"MINIMUM", ROUND(SUM(EMP_SAL),0)"SUM",ROUND(AVG(EMP_SAL),0)"AVERAGE" FROM EMPLOYEE
```

The results pane displays the output:

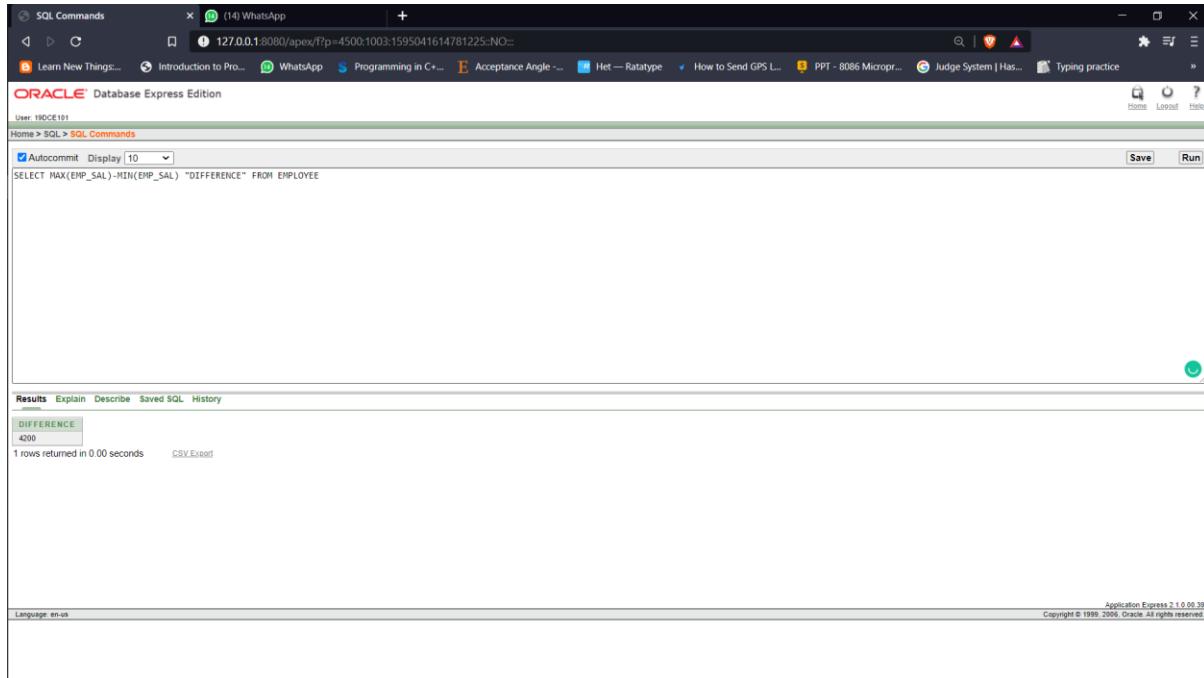
| MAXIMUM | MINIMUM | SUM | AVERAGE |
|---------|---------|-------|---------|
| 5000 | 800 | 16925 | 2419 |

1 rows returned in 0.00 seconds CSV Export

Language: en-us Application Express 2.1.0.0.39 Copyright © 1999-2006, Oracle. All rights reserved.

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

Snap-Shot:

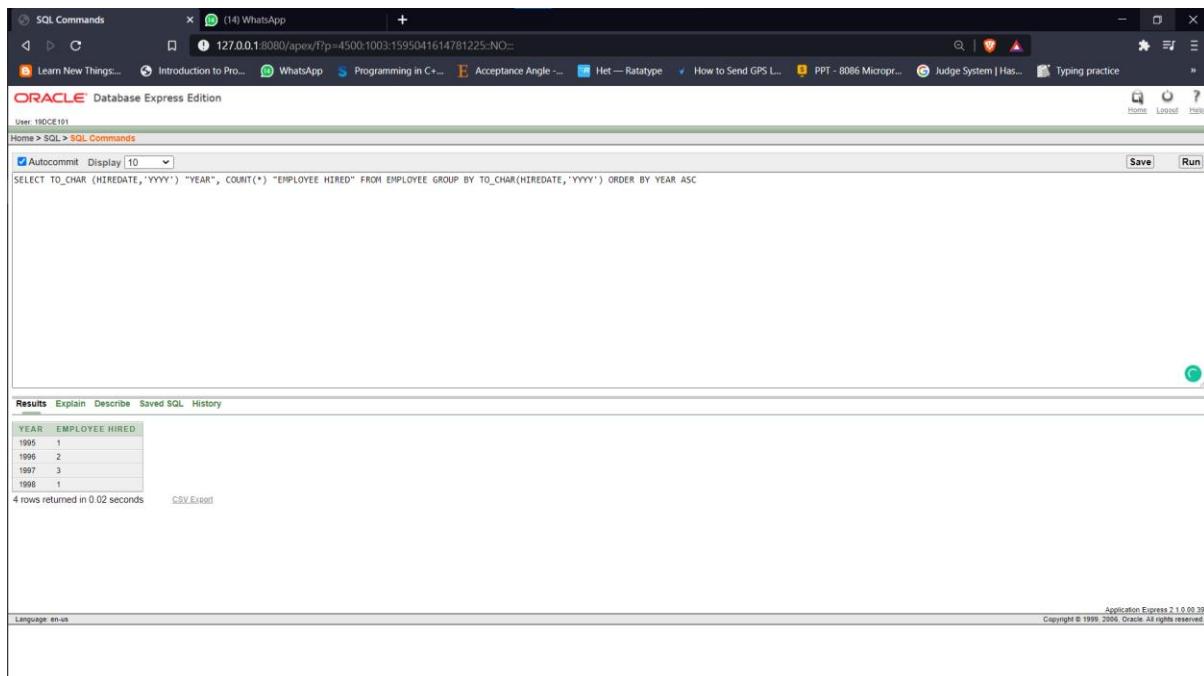


```
SQL Commands
SELECT MAX(EMP_SAL)-MIN(EMP_SAL) "DIFFERENCE" FROM EMPLOYEE
```

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, a query is entered to calculate the difference between the maximum and minimum salaries in the EMPLOYEE table. The results show a single row with the column name 'DIFFERENCE' and the value '4200'. The application version is Application Express 2.1.0.00.39.

(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

Snap-Shot:

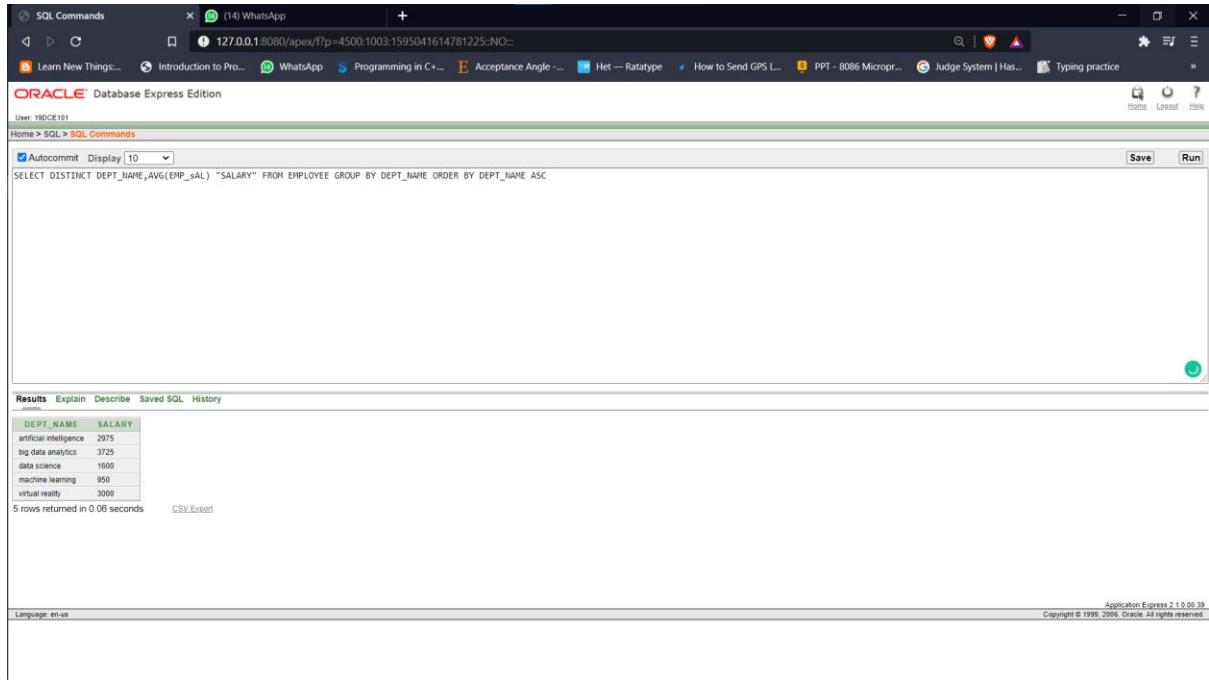


```
SQL Commands
SELECT TO_CHAR(HIREDATE,'YYYY') "YEAR", COUNT(*) "EMPLOYEE_HIRED" FROM EMPLOYEE GROUP BY TO_CHAR(HIREDATE,'YYYY') ORDER BY YEAR ASC
```

The screenshot shows the Oracle Database Express Edition interface. A query is run to group employees by their hire year and count the number of employees hired each year. The results are displayed in a table with columns 'YEAR' and 'EMPLOYEE_HIRED'. The data shows 1 employee hired in 1995, 2 in 1996, 3 in 1997, and 1 in 1998. The application version is Application Express 2.1.0.00.39.

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. A SQL command is entered in the SQL Commands window:

```
SELECT DISTINCT DEPT_NAME, AVG(EMP_SAL) "SALARY" FROM EMPLOYEE GROUP BY DEPT_NAME ORDER BY DEPT_NAME ASC
```

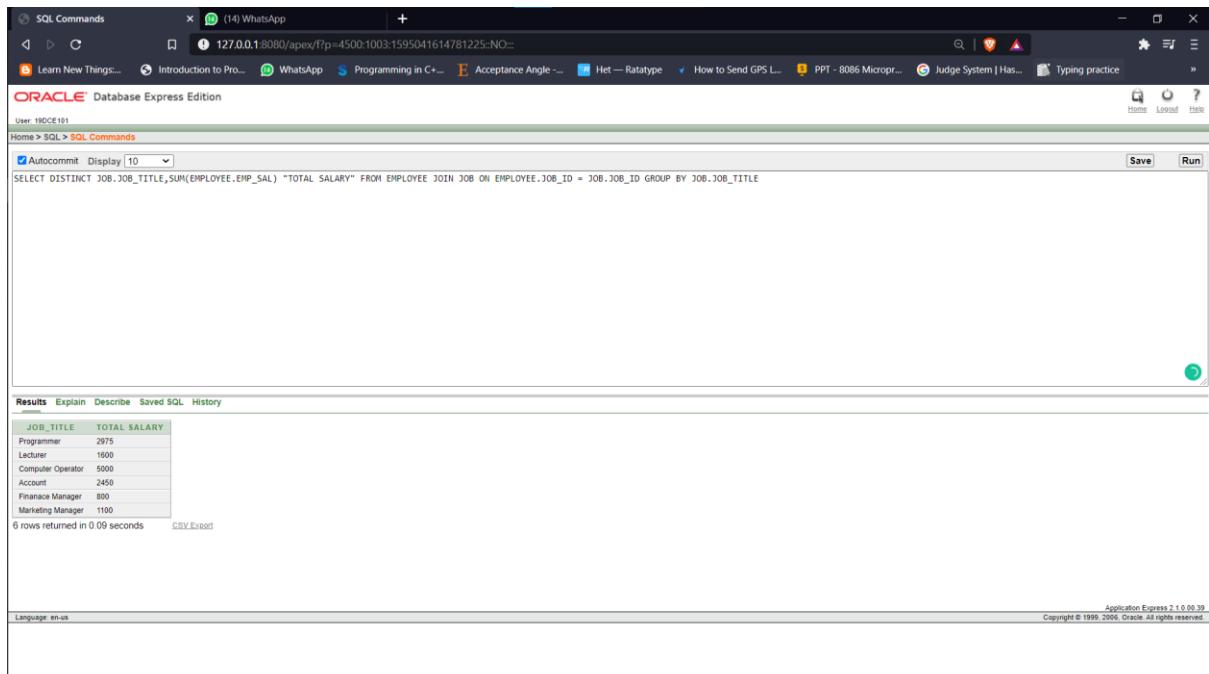
The results are displayed in a table:

| DEPT_NAME | SALARY |
|-------------------------|--------|
| artificial intelligence | 2975 |
| big data analytics | 3725 |
| data science | 1600 |
| machine learning | 950 |
| virtual reality | 300 |

5 rows returned in 0.06 seconds

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. A SQL command is entered in the SQL Commands window:

```
SELECT DISTINCT JOB.JOB_TITLE, SUM(EMPLOYEE.EMP_SAL) "TOTAL SALARY" FROM EMPLOYEE JOIN JOB ON EMPLOYEE.JOB_ID = JOB.JOB_ID GROUP BY JOB.JOB_TITLE
```

The results are displayed in a table:

| JOB_TITLE | TOTAL SALARY |
|-------------------|--------------|
| Programmer | 2975 |
| Lecturer | 1600 |
| Computer Operator | 5000 |
| Account | 2450 |
| Finance Manager | 800 |
| Marketing Manager | 1100 |

6 rows returned in 0.09 seconds

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

Snap-Shot:

```
SELECT DISTINCT DEPT_NAME, AVG(EMP_SAL) "SALARY" FROM EMPLOYEE WHERE EMP_SAL > 2000 GROUP BY DEPT_NAME
```

| DEPT_NAME | SALARY |
|-------------------------|--------|
| big data analytics | 3725 |
| artificial intelligence | 2975 |
| virtual reality | 3000 |

3 rows returned in 0.02 seconds

(10) Display the job and total salary for each job with a total salary amount exceeding 3000 and sorts the list by the total salary.

Snap-Shot:

```
SELECT DISTINCT JOB.JOB_TITLE, SUM(EMPLOYEE.EMP_SAL) "TOTAL SALARY" FROM EMPLOYEE JOIN JOB ON EMPLOYEE.EMP_SAL >= 3000 GROUP BY JOB.JOB_TITLE ORDER BY SUM(EMPLOYEE.EMP_SAL)
```

| JOB_TITLE | TOTAL SALARY |
|-------------------|--------------|
| Programmer | 8000 |
| Account | 8000 |
| Finance Manager | 8000 |
| Computer Operator | 8000 |
| Marketing Manager | 8000 |
| Lecturer | 8000 |

6 rows returned in 0.03 seconds

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

Snap-Shot:

The screenshot shows a browser window with multiple tabs open. The active tab is titled 'SQL Commands' and displays a SQL query in the Oracle Database Express Edition interface. The query is:

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

The results pane shows a single row with the value 'POWAI'. The bottom status bar indicates '1 rows returned in 0.05 seconds' and 'CSV Export'.

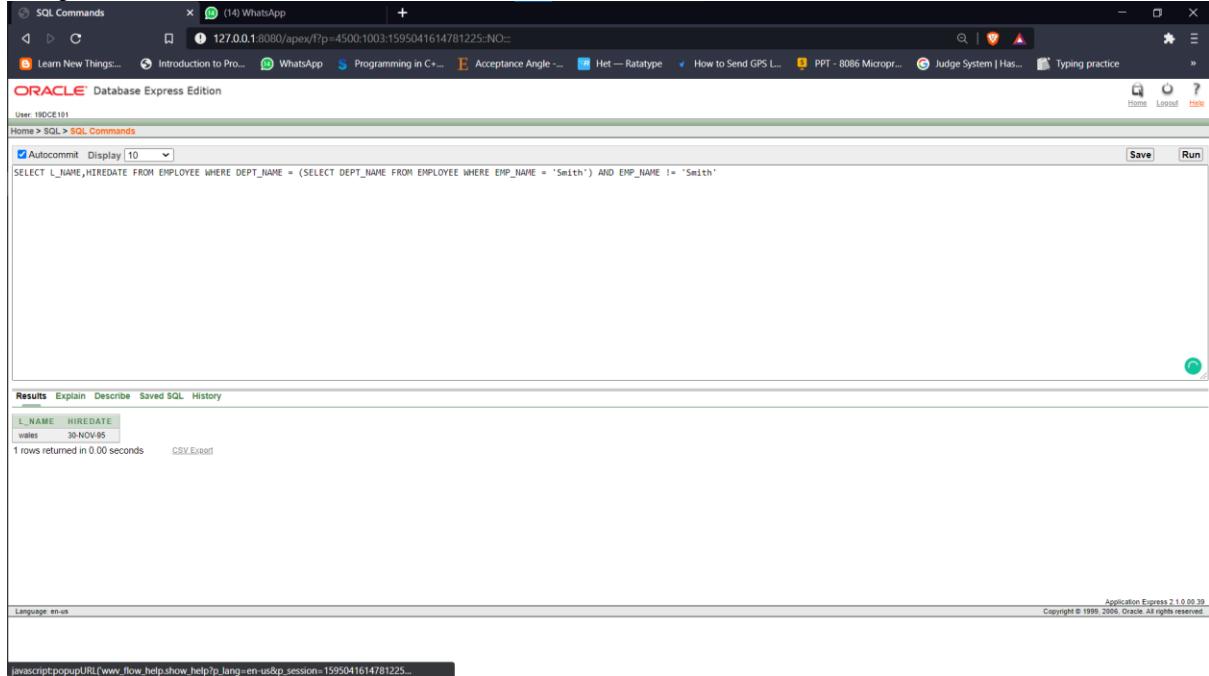
Conclusion: We Learned about Aggregating Data using Group functions.

Practical -9

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 smith. Exclude smith.

Snap-Shot:



The screenshot shows a browser window with multiple tabs open at the top. The active tab is titled "SQL Commands". The URL in the address bar is "127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO::". Below the address bar, there's a toolbar with various icons. The main content area is a SQL editor window. The SQL query entered is:

```
SELECT L_NAME,HIREDATE FROM EMPLOYEE WHERE DEPT_NAME = (SELECT DEPT_NAME FROM EMPLOYEE WHERE EMP_NAME = 'Smith') AND EMP_NAME != 'Smith'
```

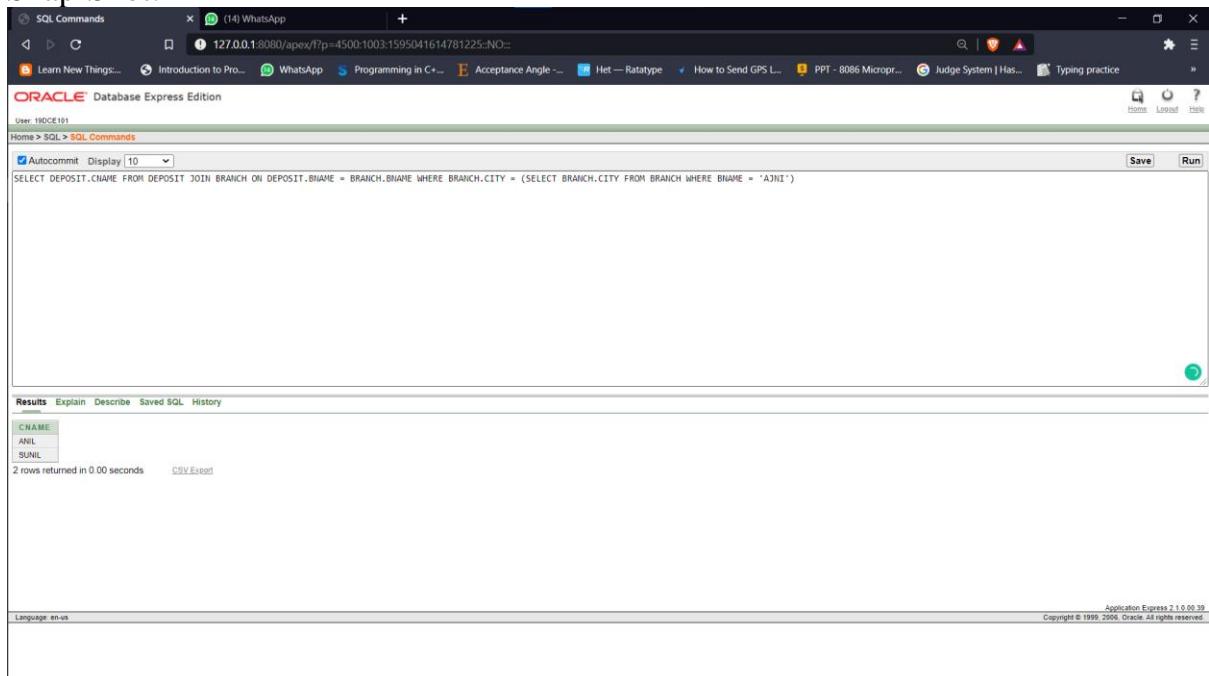
Below the query, the results are displayed in a table:

| L_NAME | HIREDATE |
|--------|-----------|
| Wales | 30-NOV-95 |

Text below the table indicates "1 rows returned in 0.00 seconds" and "CSV Export". At the bottom of the SQL editor, there are tabs for "Results", "Explain", "Describe", "Saved SQL", and "History". The status bar at the bottom right shows "Application Express 2.1.0.0.39" and "Copyright © 1999, 2006, Oracle. All rights reserved."

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

Snap-Shot:



The screenshot shows a browser window with multiple tabs open at the top. The active tab is titled "SQL Commands". The URL in the address bar is "127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO::". Below the address bar, there's a toolbar with various icons. The main content area is a SQL editor window. The SQL query entered is:

```
SELECT DEPOSIT.CNAME FROM DEPOSIT JOIN BRANCH ON DEPOSIT.BNAME = BRANCH.BNAME WHERE BRANCH.CITY = (SELECT BRANCH.CITY FROM BRANCH WHERE BNAME = 'AJNI')
```

Below the query, the results are displayed in a table:

| CNAME |
|-------|
| ANIL |
| SUNIL |

Text below the table indicates "2 rows returned in 0.00 seconds" and "CSV Export". At the bottom of the SQL editor, there are tabs for "Results", "Explain", "Describe", "Saved SQL", and "History". The status bar at the bottom right shows "Application Express 2.1.0.0.39" and "Copyright © 1999, 2006, Oracle. All rights reserved."

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

Snap-Shot:

```

SELECT DEPOSIT.ACNO,DEPOSIT.AMOUNT AS DEPOSIT_AMOUNT,BORROW.LOANNO,BORROW.AMOUNT AS BORROW_AMOUNT FROM DEPOSIT JOIN BORROW ON DEPOSIT.CNAME = BORROW.CNAME JOIN CUSTOMERS ON DEPOSIT.CNAME = CUSTOMERS.CNAME WHERE CUSTOMERS.CITY = (SELECT CITY FROM CUSTOMERS WHERE CNAME = 'PRAMOD')

```

| ACTNO | DEPOSIT_AMOUNT | LOANNO | BORROW_AMOUNT |
|-------|----------------|--------|---------------|
| 104 | 1200 | 321 | 2000 |
| 105 | 3000 | 375 | 8000 |

2 rows returned in 0.08 seconds CSV Export

(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.

Snap-Shot:

```

SELECT EMP_NO,L_NAME FROM EMPLOYEE WHERE EMP_SAL > (SELECT AVG(EMP_SAL) FROM EMPLOYEE) ORDER BY EMP_SAL ASC

```

| EMP_NO | L_NAME |
|--------|--------|
| 106 | joseph |
| 107 | tha |
| 104 | sharma |
| 105 | patel |

4 rows returned in 0.00 seconds CSV Export

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

Snap-Shot:

```
SELECT DEPOSIT.CNAME FROM DEPOSIT JOIN BRANCH ON DEPOSIT.BNAME = BRANCH.BNAME WHERE DEPOSIT.AMOUNT > 2000 AND BRANCH.CITY = (SELECT CITY FROM BRANCH WHERE BNAME = (SELECT BNAME FROM DEPOSIT WHERE CNAME = 'ANIL'))
```

The results show one row: CNAME SUNIL

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

Snap-Shot:

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

The results show no data found.

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. The SQL command window contains the following query:

```
SELECT EMPLOYEE.DEPT_NO,EMPLOYEE.DEPT_NAME,JOB.JOB_TITLE FROM EMPLOYEE JOIN JOB ON EMPLOYEE.JOB_ID = JOB.JOB_ID WHERE JOB.JOB_TITLE = 'Account'
```

The results pane displays the following data:

| DEPT_NO | DEPT_NAME | JOB_TITLE |
|---------|--------------------|-----------|
| 10 | big data analytics | Account |

1 rows returned in 0.00 seconds

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. The SQL command window contains the following query:

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

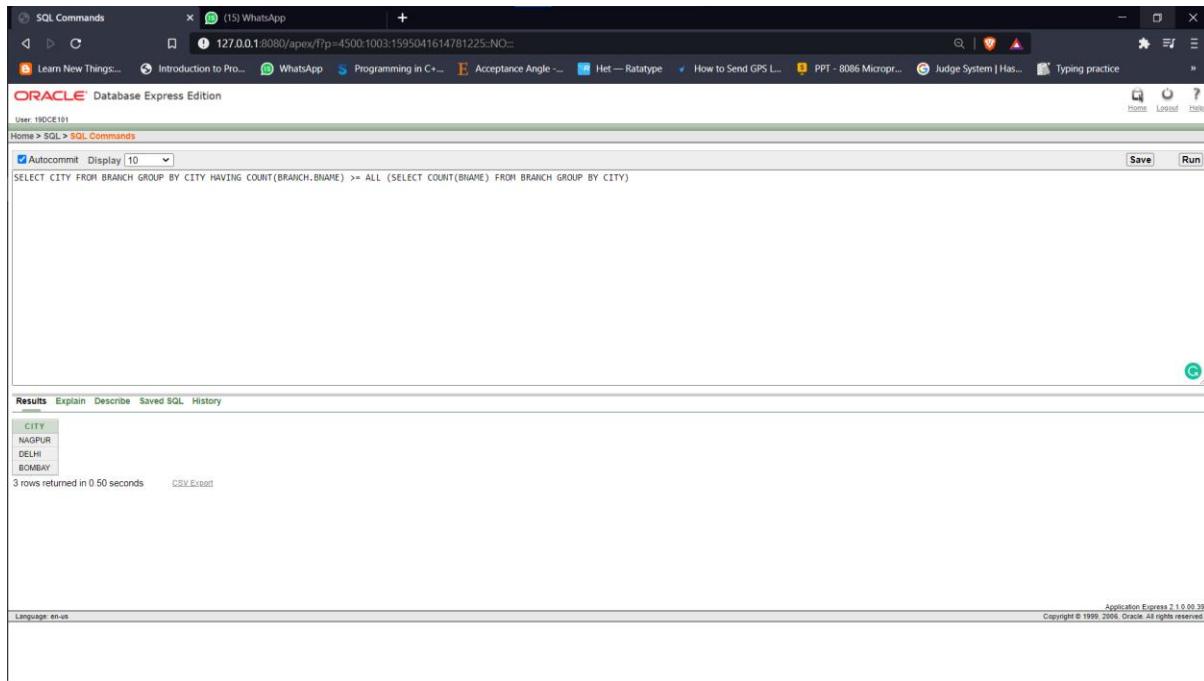
The results pane displays the following data:

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

9 rows returned in 0.01 seconds

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following query:

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

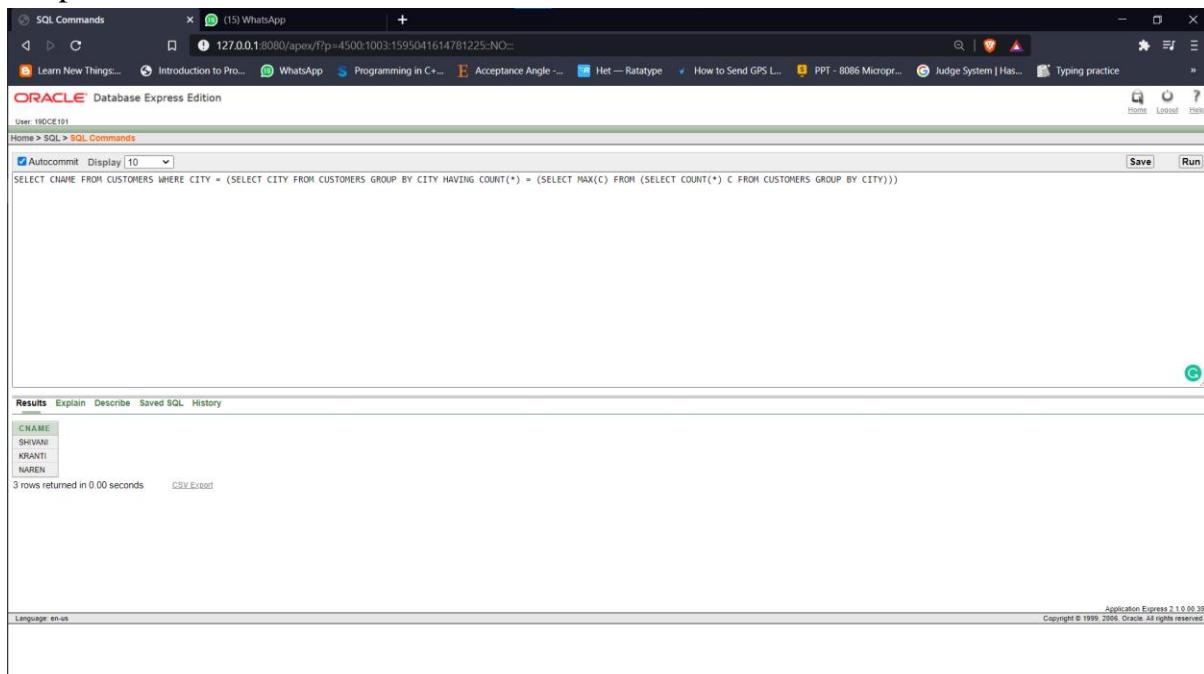
The results pane displays the following data:

| CITY |
|--------|
| NAGPUR |
| DELHI |
| BOMBAY |

3 rows returned in 0.50 seconds

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following query:

```
SELECT CNAME FROM CUSTOMERS WHERE CITY = (SELECT CITY FROM CUSTOMERS GROUP BY CITY HAVING COUNT(*) = (SELECT MAX(C) FROM (SELECT COUNT(*) C FROM CUSTOMERS GROUP BY CITY)))
```

The results pane displays the following data:

| CNAME |
|---------|
| SHIVANI |
| KRANTI |
| NAREN |

3 rows returned in 0.00 seconds

Conclusion: From This practical I learned how to perform sub-queries.

Practical -10

Manipulating Data

(1) Give 10% interest to all depositors.

Snap-Shot:

```

SQL Commands
127.0.0.1:8080/apex/r?p=4500:1003:1595041614781225:NO...
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle ... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10
UPDATE DEPOSIT SET AMOUNT = AMOUNT*1.10;
SELECT * FROM DEPOSIT
Results Explain Describe Saved SQL History
ACTNO CNAME BNAME AMOUNT ADATE
100 ANIL VRCE 1100 01-MAR-95
101 SUNI AJNI 5500 04-JAN-95
102 MEHUL KAROLBAH 3850 17-NOV-95
104 MADHURI CHANDI 1320 17-DEC-95
105 PRAMOD M.O.ROAD 3300 27-MAR-95
106 SANDIP ANDHERI 2200 31-MAR-95
107 SHIVANI VIRAR 1100 05-SEP-95
108 KRANTI NEHRU PLACE 5500 02-JUL-95
109 MINU POWAI 7700 10-AUG-95
9 rows returned in 0.05 seconds CSV Export
Language: en-us
Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

```

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

Snap-Shot:

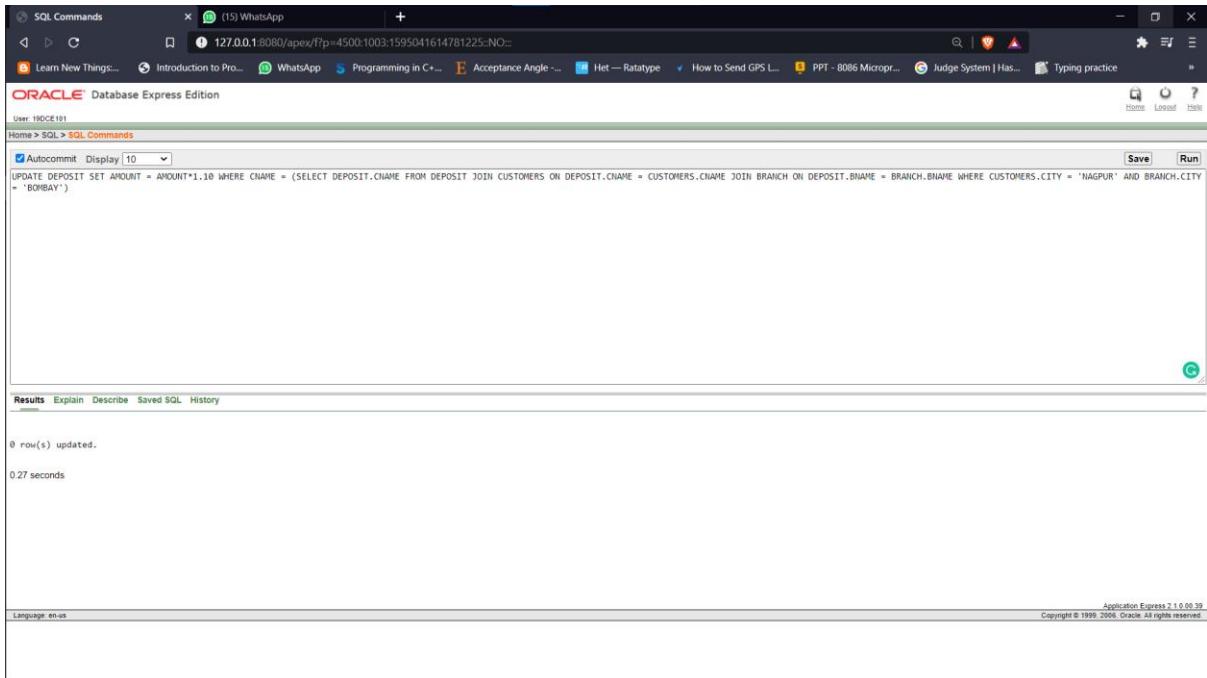
```

SQL Commands
127.0.0.1:8080/apex/r?p=4500:1003:1595041614781225:NO...
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle ... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10
UPDATE DEPOSIT SET AMOUNT = AMOUNT*1.10 WHERE BNAME = 'VRCE';
SELECT * FROM DEPOSIT
Results Explain Describe Saved SQL History
ACTNO CNAME BNAME AMOUNT ADATE
100 ANIL VRCE 1210 01-MAR-95
101 SUNI AJNI 5500 04-JAN-95
102 MEHUL KAROLBAH 3850 17-NOV-95
104 MADHURI CHANDI 1320 17-DEC-95
105 PRAMOD M.O.ROAD 3300 27-MAR-95
106 SANDIP ANDHERI 2200 31-MAR-95
107 SHIVANI VIRAR 1100 05-SEP-95
108 KRANTI NEHRU PLACE 5500 02-JUL-95
109 MINU POWAI 7700 10-AUG-95
9 rows returned in 0.00 seconds CSV Export
Language: en-us
Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

```

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition SQL Commands interface. The SQL command entered is:

```
UPDATE DEPOSIT SET AMOUNT = AMOUNT*1.10 WHERE CNAME = (SELECT DEPOSIT.CNAME FROM DEPOSIT JOIN CUSTOMERS ON DEPOSIT.CNAME = CUSTOMERS.CNAME JOIN BRANCH ON DEPOSIT.BNAME = BRANCH.BNAME WHERE CUSTOMERS.CITY = 'NAGPUR' AND BRANCH.CITY = 'BOMBAY')
```

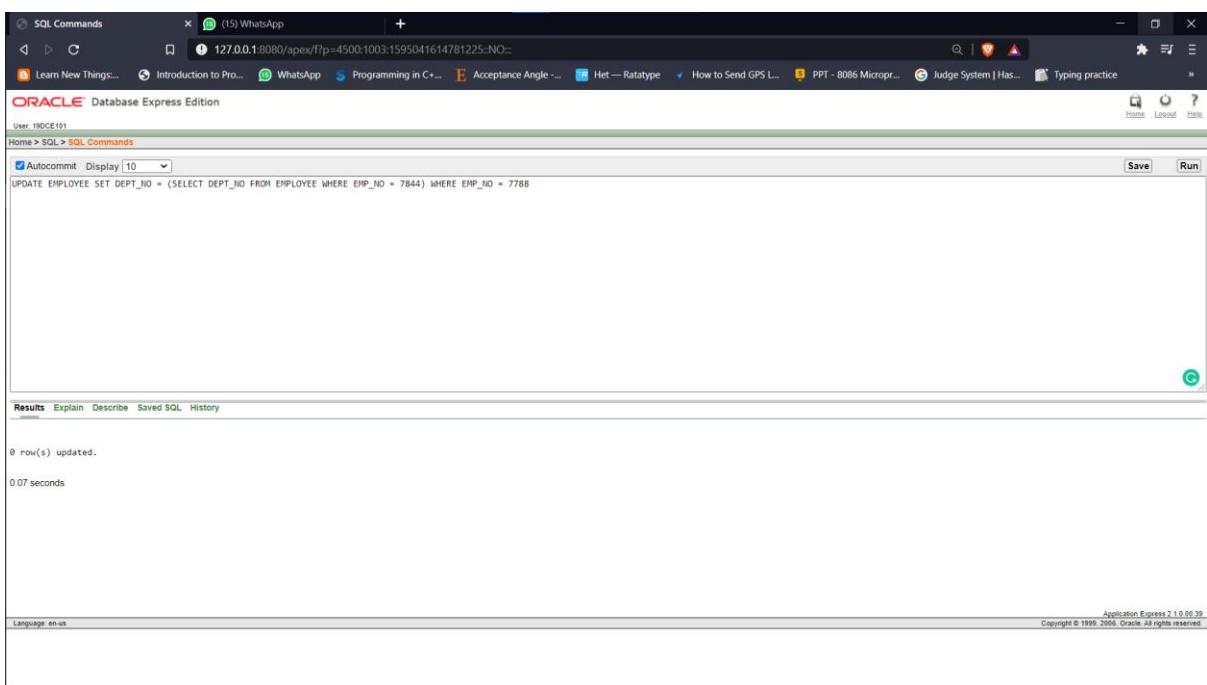
The results section shows:

- 0 row(s) updated.
- 0.27 seconds

At the bottom, it says Application Express 2.1.0.0.39 Copyright © 1999-2006, Oracle. All rights reserved.

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition SQL Commands interface. The SQL command entered is:

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

The results section shows:

- 0 row(s) updated.
- 0.07 seconds

At the bottom, it says Application Express 2.1.0.0.39 Copyright © 1999-2006, Oracle. All rights reserved.

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

Snap-Shot:

```

SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO::
Home > SQL > SQL Commands
Autocommit Display 10 Save Run
SELECT * FROM DEPOSIT
UPDATE DEPOSIT SET BNAME = 'VRCE' WHERE ACTNO = 101
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')

Results Explain Describe Saved SQL History
ACTNO CNAME BNAME AMOUNT ADATE
100 ANIL VRCE 1200 01-MAR-95
101 SUNIL VRCE 5510 04-JAN-96
102 MEHUL KAROLBAGH 3850 17-NOV-95
104 MADHURI CHANDI 1320 17-DEC-95
105 PRAMOD M.O.ROAD 3300 27-MAR-96
106 SANDIP ANDHERI 2200 31-MAR-96
107 SHIVANI VIRAR 1100 05-SEP-95
108 KRANTI NEHRU PLACE 5500 02-JUL-95
109 MINU POWAI 7700 10-AUG-95
9 rows returned in 0.00 seconds CSV Export
Language: en-us Application Express 2.1.0.0.39 Copyright © 1999, 2006, Oracle. All rights reserved.

```

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

Snap-Shot:

```

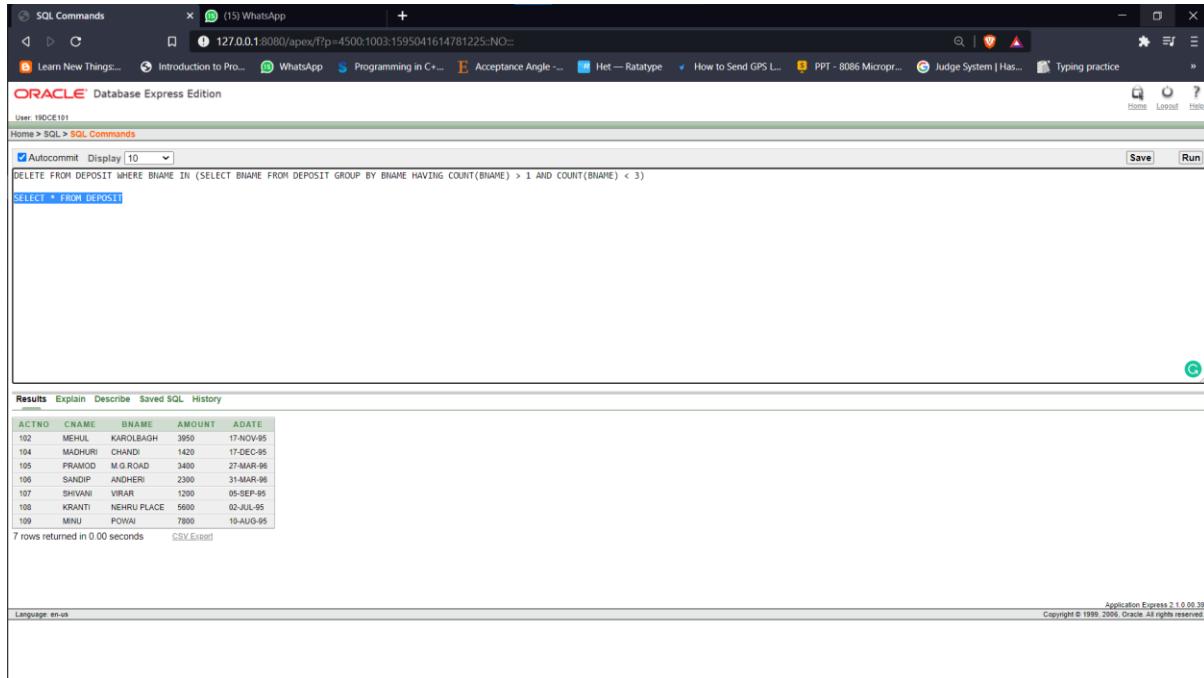
SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO::
Home > SQL > SQL Commands
Autocommit Display 10 Save Run
UPDATE DEPOSIT SET AMOUNT = AMOUNT + 100 WHERE CNAME = ANY(SELECT CNAME FROM DEPOSIT WHERE AMOUNT IN (SELECT MAX(AMOUNT) FROM DEPOSIT GROUP BY BNAME))
SELECT * FROM DEPOSIT

Results Explain Describe Saved SQL History
ACTNO CNAME BNAME AMOUNT ADATE
100 ANIL VRCE 1200 01-MAR-95
101 SUNIL VRCE 5610 04-JAN-96
102 MEHUL KAROLBAGH 3950 17-NOV-95
104 MADHURI CHANDI 1420 17-DEC-95
105 PRAMOD M.O.ROAD 3400 27-MAR-96
106 SANDIP ANDHERI 2300 31-MAR-96
107 SHIVANI VIRAR 1200 05-SEP-95
108 KRANTI NEHRU PLACE 5600 02-JUL-95
109 MINU POWAI 7800 10-AUG-95
9 rows returned in 0.00 seconds CSV Export
Language: en-us Application Express 2.1.0.0.39 Copyright © 1999, 2006, Oracle. All rights reserved.

```

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

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. The SQL command window contains the following code:

```
DELETE FROM DEPOSIT WHERE BNAME IN (SELECT BNAME FROM DEPOSIT GROUP BY BNAME HAVING COUNT(BNAME) > 1 AND COUNT(BNAME) < 3)
```

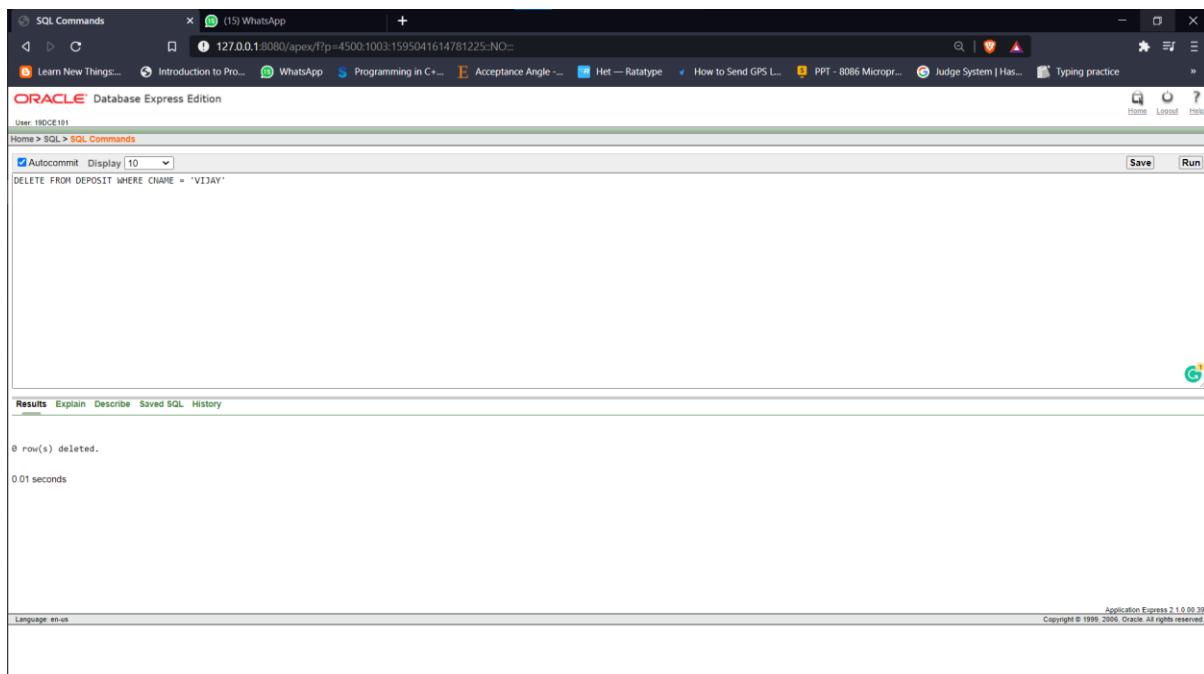
Below the command, the results section displays a table with columns ACTNO, CNAME, BNAME, AMOUNT, and ADATE. The data is as follows:

| ACTNO | CNAME | BNAME | AMOUNT | ADATE |
|-------|---------|-------------|--------|-----------|
| 102 | MEHUL | KAROLBAGH | 3950 | 17-NOV-95 |
| 104 | MADHUR | CHANDA | 1420 | 17-DEC-95 |
| 105 | PRAMOD | M G ROAD | 3400 | 27-MAR-96 |
| 106 | SANDIP | ANDHERI | 2300 | 31-MAR-96 |
| 107 | SHIVANI | VIRAR | 1200 | 05-SEP-95 |
| 108 | KRANTI | NEHRU PLACE | 5600 | 02-JUL-95 |
| 109 | MINU | POWAI | 7800 | 10-AUG-95 |

7 rows returned in 0.00 seconds

(8) Delete deposit of vijay.

Snap-Shot:



The screenshot shows the Oracle Database Express Edition interface. The SQL command window contains the following code:

```
DELETE FROM DEPOSIT WHERE CNAME = 'VIJAY'
```

The results section shows the message:

0 row(s) deleted.
0.01 seconds

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

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. The URL in the address bar is 127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO:::. The page title is "SQL Commands". The SQL command entered is "DELETE FROM BORROW WHERE AMOUNT < 1000". The results section shows "0 row(s) deleted." and "0.05 seconds". The bottom right corner of the interface displays "Application Express 2.1.0.0.39" and "Copyright © 1999, 2006, Oracle. All rights reserved."

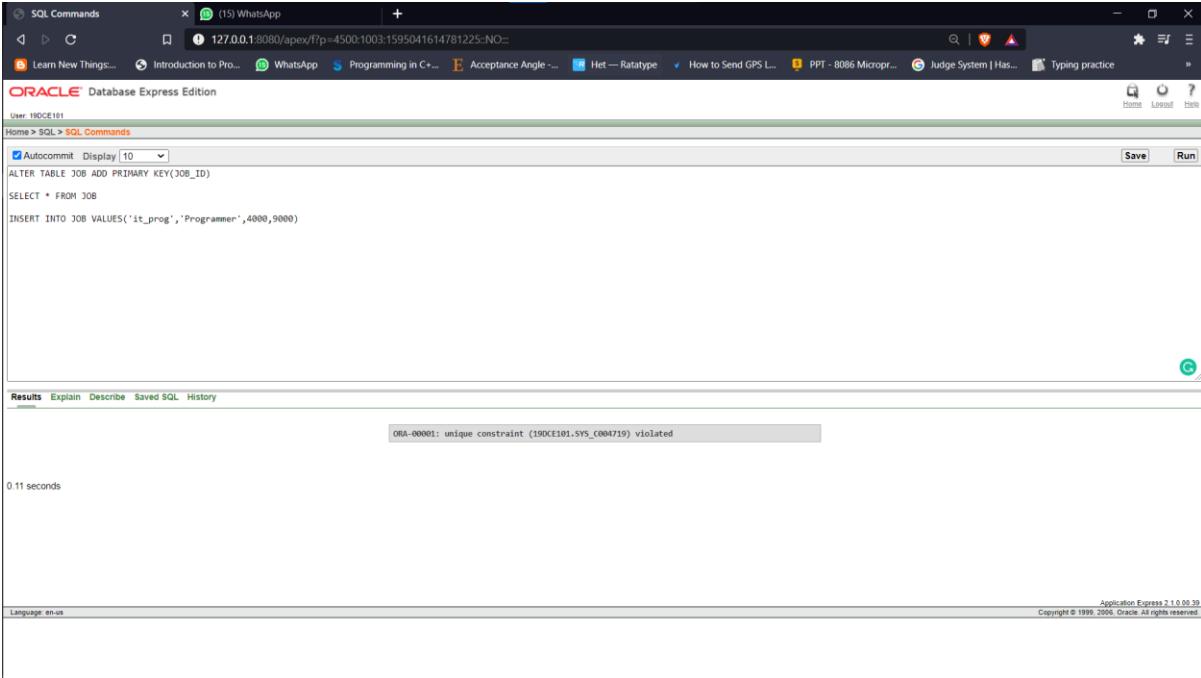
Conclusion: From this practical I learned how to manipulate data.

Practical -11

Add and Remove constraint

(1) Add primary key constraint on job_id in job table.

Snap-Shot:



The screenshot shows a browser window with the URL `127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO::`. The page title is "SQL Commands". The SQL command entered is:

```
ALTER TABLE JOB ADD PRIMARY KEY(JOB_ID)
SELECT * FROM JOB
INSERT INTO JOB VALUES('it_prog','Programmer',4000,9000)
```

The results section shows an error message: "ORA-00001: unique constraint (19DCE101.SYS_C004719) violated".

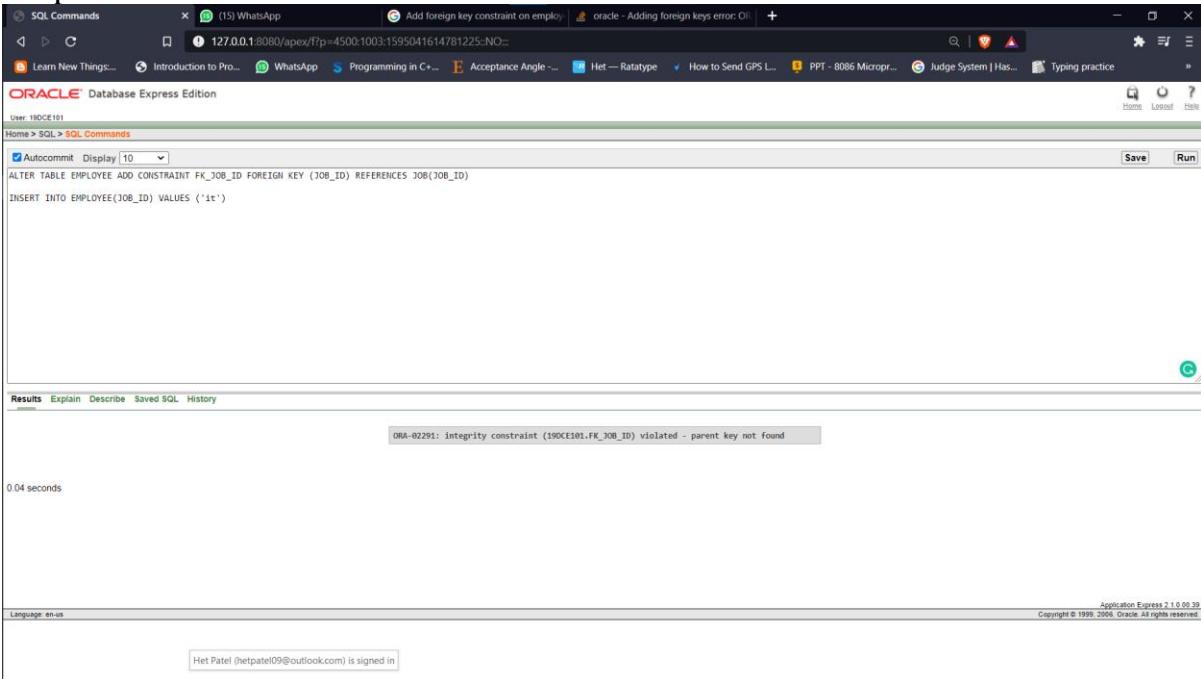
Time taken: 0.11 seconds

Language: en-us

Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

(2) Add foreign key constraint on employee table referencing job table.

Snap-Shot:



The screenshot shows a browser window with the URL `127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO::`. The page title is "Add foreign key constraint on emplo... oracle - Adding foreign keys error: ORA-02291". The SQL command entered is:

```
ALTER TABLE EMPLOYEE ADD CONSTRAINT FK_JOB_ID FOREIGN KEY (JOB_ID) REFERENCES JOB(JOB_ID)
INSERT INTO EMPLOYEE(JOB_ID) VALUES ('it')
```

The results section shows an error message: "ORA-02291: integrity constraint (19DCE101.FK_JOB_ID) violated - parent key not found".

Time taken: 0.04 seconds

Language: en-us

Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

Het Patel (hetpatel09@outlook.com) is signed in

(3) Add composite primary key on lock table (lock table does not exist, while creating table add composite key)

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. In the SQL editor, the following code is entered:

```
CREATE TABLE LOCK11(COMpany VARCHAR2(30), Model VARCHAR2(30), PRIMARY KEY(Company,Model))
SELECT * FROM LOCK11
INSERT INTO LOCK11(Company,Model) VALUES('Euro','654-5259x')
INSERT INTO LOCK11(Company,Model) VALUES('Euro','654-5259x')
```

The results pane displays the error message:

ORA-00001: unique constraint (19DCE101.SYS_C004726) violated

Execution time: 0.00 seconds.

Language: en-us

Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

(4) Remove primary key constraint on job_id

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. In the SQL editor, the following code is entered:

```
ALTER TABLE EMPLOYEE DROP CONSTRAINT FK_JOB_ID
ALTER TABLE JOB DROP PRIMARY KEY
```

The results pane displays the message:

Table dropped.

Execution time: 0.79 seconds.

Language: en-us

Application Express 2.1.0.0.39
Copyright © 1999, 2006, Oracle. All rights reserved.

(5) Remove foreign key constraint on employee table

Snap-Shot:

The screenshot shows the Oracle Database Express Edition SQL Commands interface. In the top navigation bar, there are several tabs: SQL Commands, WhatsApp, Add foreign key constraint on employe..., Sql Server Drop Foreign Key - javatpc..., and others. Below the tabs, the URL is 127.0.0.1:8080/apex/f?p=4500:1003:159504161478122::NO:. The main area is titled "ORACLE Database Express Edition" and shows the SQL command: "ALTER TABLE EMPLOYEE DROP CONSTRAINT FK_JOB_ID". Below the command, the results are displayed: "Table dropped." and "0.79 seconds". At the bottom right, it says "Application Express 2.1.0.0.39 Copyright © 1999, 2006, Oracle. All rights reserved."

Conclusion: From this practical I learned how to remove and add constraints on a table.

Practical -12

(12.1) Data Dictionary

Snap-shot:

The screenshot shows an Excel spreadsheet titled "Data - Excel" with a table named "Sheet1". The table has columns for Field ID, Data type, Field Name, Examples, and Constraints. The rows contain various database field definitions, such as emp_id (number), emp_name (text), F_name (text), L_name (text), Designation (text), joining_date (date), year_of_experience (number), Cleaning_assig_Id (text), Room_id (number), Room_Type (text), Occupation (Date / Time), Credit_Card (number), and Documents (text). The table also includes notes for Room_Type and Cleaning_assig_Id.

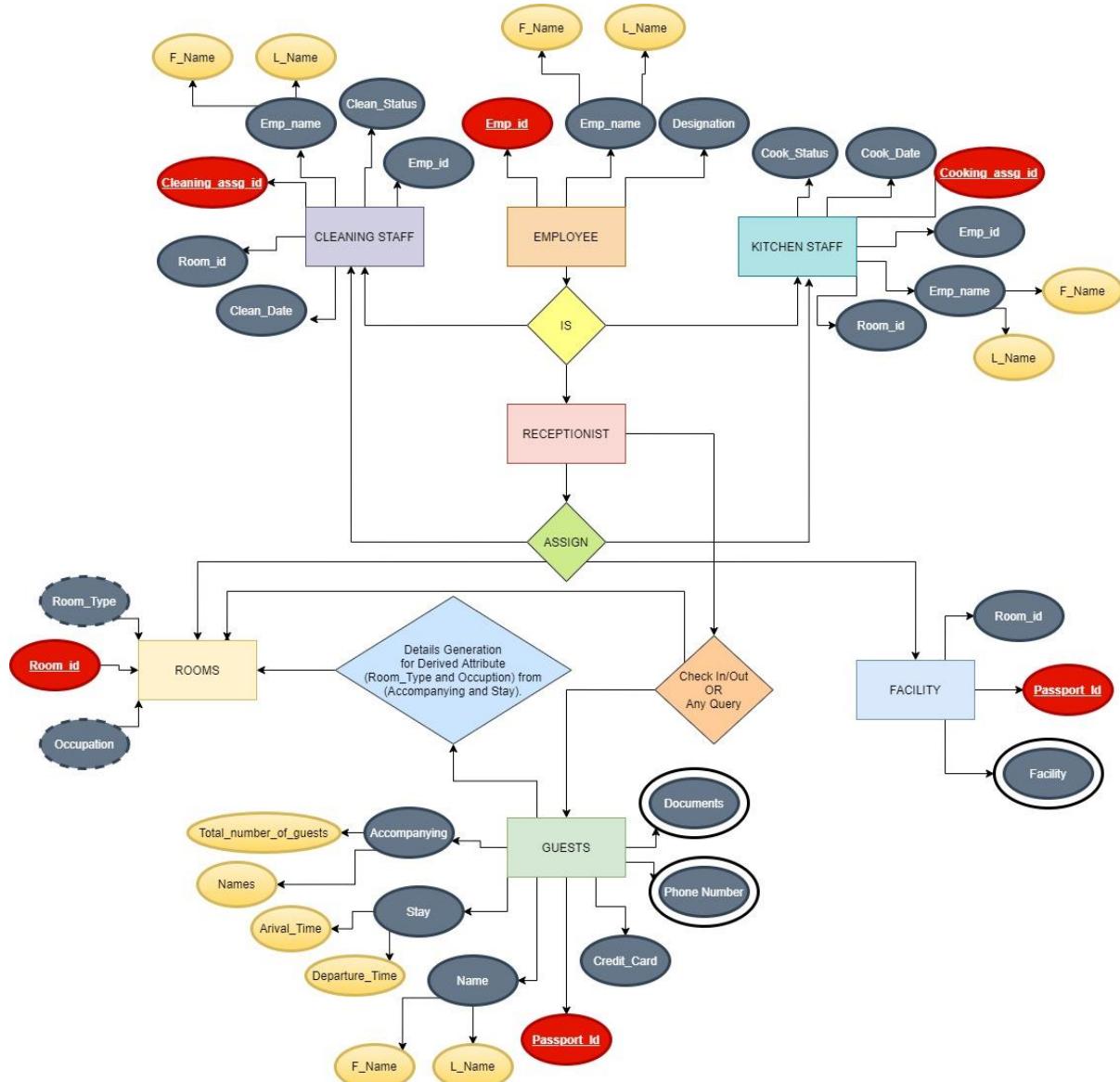
| Field ID | Data type | Field Name | Examples | Constraints |
|----------|-------------|---|------------------|---------------------|
| 2 | number | Employee Unique Id | 101 | PK |
| 3 | text | Employee Name (F_name + L_name) | Het Patel | Non Null |
| 4 | text | First Name | Het | Non Null |
| 5 | text | Last Name | Patel | Non Null |
| 6 | text | Designation of an employee | Cleaner | Non Null |
| 7 | date | Date of Joining | 28-01-2002 | Non Null |
| 8 | number | Current year - joining year | 18 | Non Null |
| 9 | text | Format :- Date(6) + Emp_id(6)+Room_id(4) | 200120021011234 | PK |
| 10 | number | Unique Room Number | 1234 | PK |
| | | It can be either SR(Single Room) or DR (Double Room) or TR(Triple Room) Also it is Derived Attribute | SR | Non Null |
| 11 | | | | |
| 12 | Date / Time | Room will be occupied (Departure Time-Arrival Time) | 48 | Non Null |
| 13 | number | Credit Card Number and its details | 1234 4567 7890 | Unique And Not Null |
| 14 | text | More details about guest like their address proof, driving licence or adhar card | GJ02 20200024042 | Not Null |

The screenshot shows an Excel spreadsheet titled "Data - Excel" with a table named "Sheet1". This table continues the list of fields from the previous table, including Passport_id, Phone_number, Arrival_Time, Depature_time, Name, Accompanying, Clean_Date, Clean_Status, Facility, Cooking_assign_id, Cook_Date, Cook_Status, and Stay. It includes detailed notes for the Accompanying and Facility fields.

| A | B | C | D | E | F |
|----|-------------|----|--|---|---------------------|
| 15 | number | 9 | PassPort Number of guest | 123456789 | PK |
| 16 | number | 10 | Phone number of Guest | 8153520325 | Unique And Not Null |
| 17 | Date / Time | 10 | Arrival date with time | 01/01/2007 12:30pm | Not Null |
| 18 | Date / Time | 10 | Depature date with time | 02/01/2007 12:30pm | Not Null |
| 19 | text | 20 | Guest Name (F_name + L_name) | Priyank Shah | Not Null |
| 20 | 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" | 6 Het, Dhruv, Vishwas, Nirav, Purav, Vrusang | Not Null |
| 21 | Date / Time | 10 | Cleaning Date and Time | 01/01/2007 12:00pm | Not Null |
| 22 | text | 10 | Its describes current cleaning status of an assigned room | Cleaning Done ! | Not Null |
| 23 | text | 20 | It can have multiple activites like (Gym and Swimming) according to guest package. | Gym, Game Zone and Swimming Pool | Not Null |
| 24 | text | 16 | Format :- Date(6) + Emp_id(6)+Room_id(4) | 10120071274321 | PK |
| 25 | Date / Time | 10 | Cooking Date and Time | 01/01/2007 12:30pm | Not Null |
| 26 | text | 10 | Its describes current cooking status of an assigned room | Food is ready ! | Not Null |
| 27 | Date / Time | 10 | Guest Stay(Depature Time - Arrival Time) | 48 hours | Not Null |
| 28 | | | | | |

(12.2) E-R Diagram.

Snap-shot:



Conclusion: From This Practical I Learned how to make an E-R Diagram.

Practical -13

To perform basic PL/SQL blocks

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

Code:

```

DECLARE
    x number;
    y number;
    z number;
    total number;
    average number;

BEGIN
    x:= 10;
    y:= 20;
    z:= 30;
    total:= x+y+z;
    average:= total/3;
    DBMS_OUTPUT.PUT_LINE('Sum is'||total);
    DBMS_OUTPUT.PUT_LINE('Average is'||average);
END;

```

Snap-Shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following PL/SQL block is entered:

```

DECLARE
    x number;
    y number;
    z number;
    total number;
    average number;

BEGIN
    x:= 10;
    y:= 20;
    z:= 30;
    total:= x+y+z;
    average:= total/3;
    DBMS_OUTPUT.PUT_LINE('Sum is'||total);
    DBMS_OUTPUT.PUT_LINE('Average is'||average);
END;

```

After executing the block, the results pane displays:

```

Sum is 60
Average is 20
Statement processed.
0.25 seconds

```

At the bottom of the interface, it says "Language: en-us" and "Application Express 2.1.0.0939".

Conclusion: From this practical I learned How to write code in PL/SQL.

Practical -14

To perform the concept of loop

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

Code:

Basic Loop

```
DECLARE
    x number := 5;
    fact number :=1;

BEGIN
    LOOP
        fact:= fact * x;
        x:= x-1;
        IF x<2 THEN
            exit;
        END IF;
    END LOOP;
    dbms_output.put_line('Factorial is: ' || fact);
END;
```

While Loop

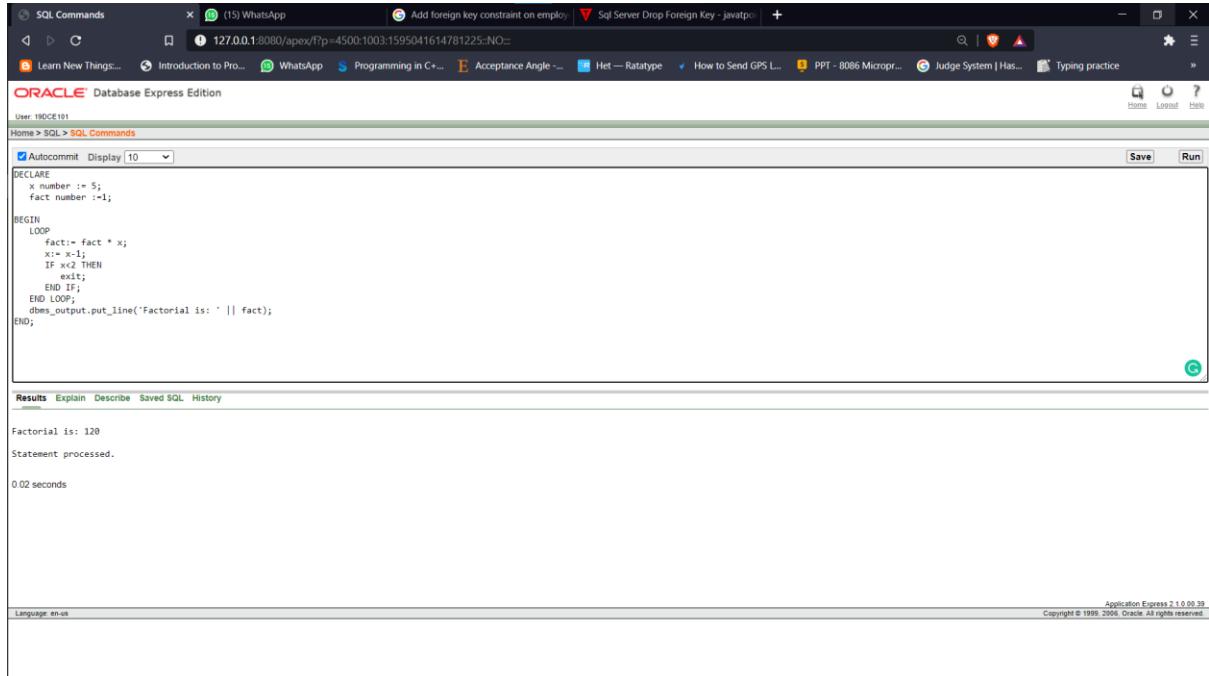
```
DECLARE
    x number := 5;
    fact number :=1;

BEGIN
    WHILE x>=2 LOOP
        fact:= fact * x;
        x:= x-1;
    END LOOP;

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

Snap-Shot:

Basic Loop



The screenshot shows a SQL Commands window in Oracle Database Express Edition. The code is as follows:

```

DECLARE
    x number := 5;
    fact number :=1;
BEGIN
    LOOP
        fact:= fact * x;
        x:= x-1;
        IF x=1 THEN
            EXIT;
        END IF;
    END LOOP;
    dbms_output.put_line('Factorial is: ' || fact);
END;

```

The results section shows:

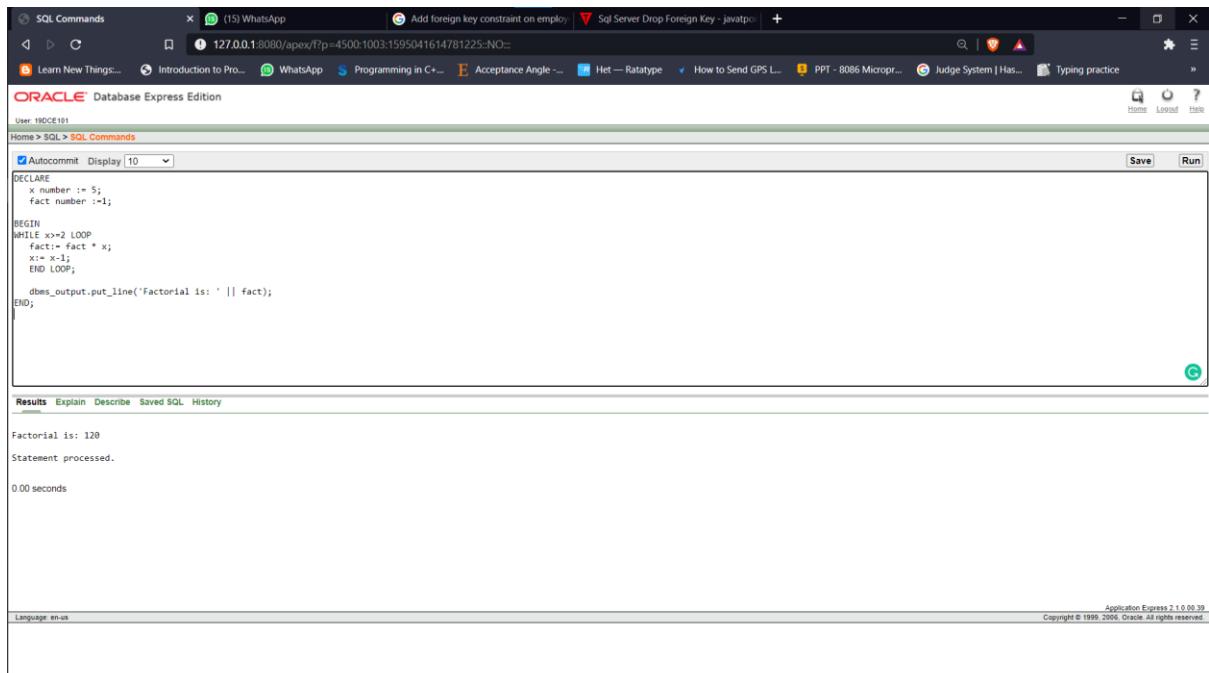
```

Factorial is: 120
Statement processed.
0.02 seconds

```

At the bottom right, it says "Application Express 2.1.0.09.39 Copyright © 1999, 2006, Oracle. All rights reserved."

While Loop



The screenshot shows a SQL Commands window in Oracle Database Express Edition. The code is as follows:

```

DECLARE
    x number := 5;
    fact number :=1;
BEGIN
    WHILE x>=2 LOOP
        fact:= fact * x;
        x:= x-1;
    END LOOP;
    dbms_output.put_line('Factorial is: ' || fact);
END;

```

The results section shows:

```

Factorial is: 120
Statement processed.
0.00 seconds

```

At the bottom right, it says "Application Express 2.1.0.09.39 Copyright © 1999, 2006, Oracle. All rights reserved."

Conclusion: From this practical I learned How to write simple loop & while loop in PL/SQL.

Practical -15

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:

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

```
DECLARE
```

```
EMP1 EMP%ROWTYPE;
```

```
CNT NUMBER:=1;
```

```
CNT_TOTAL NUMBER;
```

```
STARXSX EMP1.STARS%TYPE;
```

```
BEGIN
```

```
CNT_TOTAL:= 7;
```

```
WHILE CNT<=CNT_TOTAL LOOP
```

```
    SELECT * INTO EMP1 FROM EMP WHERE EMP_NO = 100+CNT;
```

```
    IF EMP1.EMP_SAL > 3000 THEN
        STARXSX := '* * * *';
```

```
    ELSIF EMP1.EMP_SAL >2000 THEN
        STARXSX := '* * * *';
```

```

ELSIF EMP1.EMP_SAL >1000 THEN
  STARSX := '* *';

ELSE
  STARSX := '*';

END IF;

UPDATE EMP SET STARS = STARSX
WHERE EMP_NO = EMP1.EMP_NO;

CNT:=CNT+1;

END LOOP;

END;

```

Snap-Shot:

```

SQL Commands
127.0.0.1:8080/apex?p=4500:1003:159504161478125.NO=
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle ... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
AutoCommit Display 10 Save Run
CREATE TABLE EMP AS (SELECT * FROM EMPLOYEE)
ALTER TABLE EMP ADD (STARS VARCHAR(50))
SELECT * FROM EMP
DECLARE
EMP1 EMP%ROWTYPE;
CNT NUMBER:=1;
CNT_TOTAL NUMBER;
STARSX EMP1.STARS%TYPE;
BEGIN
CNT_TOTAL:= 7;
Results Explain Describe Saved SQL History
EMP_NO EMP_NAME EMP_SAL EMP_COMM DEPT_NO L_NAME DEPT_NAME JOB_ID LOCATION MANAGER_ID HIREDATE PHONE NEW_SALARY INCREASE STARS
101 Smith 800 - 10 shah machine learning f_mgr toronto 105 06-AUG-96 - 920 120 *
102 Snehal 1600 300 25 pulta data science lec las vegas - 14-AUG-96 - 1840 240 **
103 RAMESH 1100 500 20 valles machine learning ms_mgr ontario 105 30-NOV-95 - 1265 165 **
104 Aman 3000 - 10 sharma virtual reality comp_op mexico 12 02-OCT-97 - 3450 450 ***
105 Anita 5000 50000 10 patel big data analytics comp_op germany 107 01-JAN-98 - 5750 750 ****
106 Sneha 2450 24500 10 joseph big data analytics f_acc melbourne 105 26-SEP-97 - 2817.5 367.5 ***
107 Anamika 2975 - 30 jha artificial intelligence it_prog new york - 15-JUL-97 - 3421.25 446.25 ****
7 rows returned in 0.00 seconds CSV Export
Language: en-us Application Express 2.1.0.0.39 Copyright © 1999, 2006, Oracle. All rights reserved.

```

Conclusion: From this practical I learned how to update columns through PL/SQL.

Practical -16

To perform the concept of cursor

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

Code:

DECLARE

EMP1 EMP%ROWTYPE;
CNT NUMBER:=1;
CNT1 NUMBER:=7;

BEGIN

DBMS_OUTPUT.PUT_LINE('EMP_NO' || 'EMP_NAME' || 'EMP_SAL' ||
'EMP_COMM' || 'DEPT_NO' || 'L_NAME' || 'DEPT_NAME' || 'JOB_ID' || 'LOCATION'
|| 'MANAGER_ID' || 'HIREDATE');

WHILE CNT < CNT1 LOOP

SELECT * INTO EMP1 FROM EMP WHERE EMP_NO = 100+CNT;

DBMS_OUTPUT.PUT_LINE(EMP1.EMP_NO || ' ' || EMP1.EMP_NAME || ' ' ||
EMP1.EMP_SAL || ' ' || EMP1.EMP_COMM || ' ' || EMP1.DEPT_NO || ' ' ||
EMP1.L_NAME || ' ' || EMP1.DEPT_NAME || ' ' || EMP1.JOB_ID || ' ' ||
EMP1.LOCATION || ' ' || EMP1.MANAGER_ID || ' ' || EMP1.HIREDATE);

CNT:= CNT+1;

END LOOP;

END;

Snap-shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands tab, there is a PL/SQL block that concatenates employee information into a single string. The results tab shows the concatenated output for all employees in the emp table.

```

SQL Commands
127.0.0.1:8080/apex/r?p=4500:1003:1595041614781225:NO:=
Learn New Things... Introduction to Pro... WhatsApp Add foreign key constraint on emplo... plsql - PL/SQL how to add stars to a v...
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display | 10 | Save | Run
CNT1 NUMBER:=7;
BEGIN
DBMS_OUTPUT.PUT_LINE( 'EMP_NO ' || 'EMP_NAME ' || 'EMP_SAL ' || 'EMP_COMM ' || 'DEPT_NO ' || 'L_NAME ' || 'DEPT_NAME ' || 'JOB_ID ' || 'LOCATION ' || 'MANAGER_ID ' || 'HIREDATE ');
WHILE CNT < CNT1 LOOP
SELECT * INTO EMP1 FROM EMP WHERE EMP_NO = 100+CNT;
DBMS_OUTPUT.PUT_LINE( EMP1.EMP_NO || ' ' || EMP1.EMP_NAME || ' ' || EMP1.EMP_SAL || ' ' || EMP1.EMP_COMM || ' ' || EMP1.DEPT_NO || ' ' || EMP1.L_NAME || ' ' || EMP1.DEPT_NAME || ' ' );
CNT:= CNT+1;
END LOOP;
END;

```

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE |
|--------|----------|---------|----------|---------|--------|--------------------|---------|-----------|------------|-----------|
| 101 | Smith | 800 | | 10 | shah | machine learning | fi_mngr | Toronto | 105 | 09-AUG-96 |
| 102 | Snehal | 1600 | 300 | 25 | gupta | data science | lec | Las Vegas | 105 | 14-MAR-96 |
| 103 | Rakesh | 1100 | 500 | 20 | ramesh | data mining | mk_mngr | Ontario | 105 | 30-NOV-95 |
| 104 | Aman | 3000 | | 10 | sharma | virtual reality | comp_op | Mexico | 12 | 02-OCT-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 |

Statement processed.
0.02 seconds

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

Code:

```

DECLARE
EMP1 EMP%ROWTYPE;
X EMP.DEPT_NO%TYPE:= :X;

CURSOR C1 IS SELECT EMP_NAME,EMP_SAL,MANAGER_ID,L_NAME FROM EMP
WHERE DEPT_NO = X;

BEGIN

DBMS_OUTPUT.PUT_LINE( 'EMP_NAME ' || 'EMP_SAL ' || 'MANAGER_ID ' );

FOR EMP1 IN C1 LOOP

DBMS_OUTPUT.PUT_LINE( EMP1.EMP_NAME || ' ' || EMP1.EMP_SAL || ' ' || EMP1.MANAGER_ID);

END LOOP;

DBMS_OUTPUT.PUT_LINE('-----');

```

FOR EMP1 IN C1 LOOP

```
IF EMP1.EMP_SAL < 1000 THEN
  DBMS_OUTPUT.PUT_LINE(EMP1.L_NAME||' Is due for a Raise.');
```

ELSE

```
  DBMS_OUTPUT.PUT_LINE(EMP1.L_NAME||' Is not due for a Raise.');
```

END IF;

END LOOP;

END;

Snap-shot:

```

SQL Commands
127.0.0.1:8080/apex?p=4500:1003:159504161478125..NO...
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle -... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit: Display: 10 Save Run
END LOOP;
DBMS_OUTPUT.PUT_LINE('-----');
FOR EMP1 IN C1 LOOP
  IF EMP1.EMP_SAL < 1000 THEN
    DBMS_OUTPUT.PUT_LINE(EMP1.L_NAME||' Is due for a Raise.');
  ELSE
    DBMS_OUTPUT.PUT_LINE(EMP1.L_NAME||' Is not due for a Raise.');
  END IF;
END LOOP;
END;

Results Explain Describe Saved SQL History
EMPLOYEE_ID EMP_SAL LAST_NAME
Smith          800       105
Aman          3000      12
Anita          5000      107
Smeha         2450       105
-----
sharma        1500       105
sharma Is due for a Raise.
sharma Is not due for a Raise.
patel Is not due for a Raise.
joseph Is not due for a Raise.
Statement processed.

0.00 seconds
Language: en-us
Application Express 2.1.0.09.39
Copyright © 1999, 2006, Oracle. All rights reserved.

```

Conclusion: From this Practical I learned how to perform the concept of cursor.

Practical -17

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.

Code:

STEP 1 : CREATE BACKUP TABLE

```
CREATE TABLE SALARY(
    EMP_NO NUMBER,
    EMP_SAL    NUMBER
);
```

STEP 2: ADD TRIGGER

```
CREATE OR REPLACE TRIGGER COPY_SALARY
BEFORE UPDATE
ON EMPLOYEE4
FOR EACH ROW
```

DECLARE

```
EMP_SAL NUMBER;
EMP_NO NUMBER;
```

```
BEGIN
EMP_SAL := :OLD.EMP_SAL;
EMP_NO := :OLD.EMP_NO;
INSERT INTO SALARY
VALUES (EMP_NO,EMP_SAL);
END;
```

STEP 3: UPDATE VALUE IN EMP WHERE DEPT_NO = 10

```
UPDATE EMPLOYEE SET EMP_SAL = 3500 WHERE DEPT_NO = 10
```

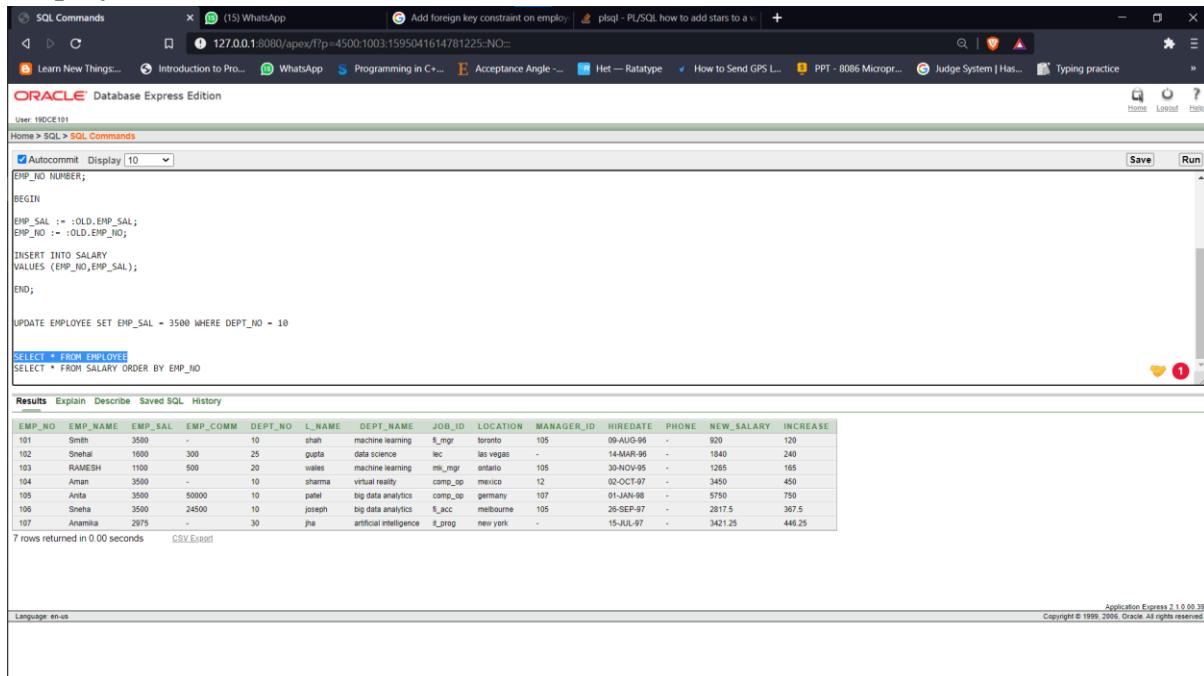
STEP 4: AFTER UPDATE CHECK VALUES IN BOTH TABLES

```
SELECT * FROM EMPLOYEE
```

```
SELECT * FROM SALARY ORDER BY EMP_NO
```

Snap-shots:

Employee Table After Command Execution:



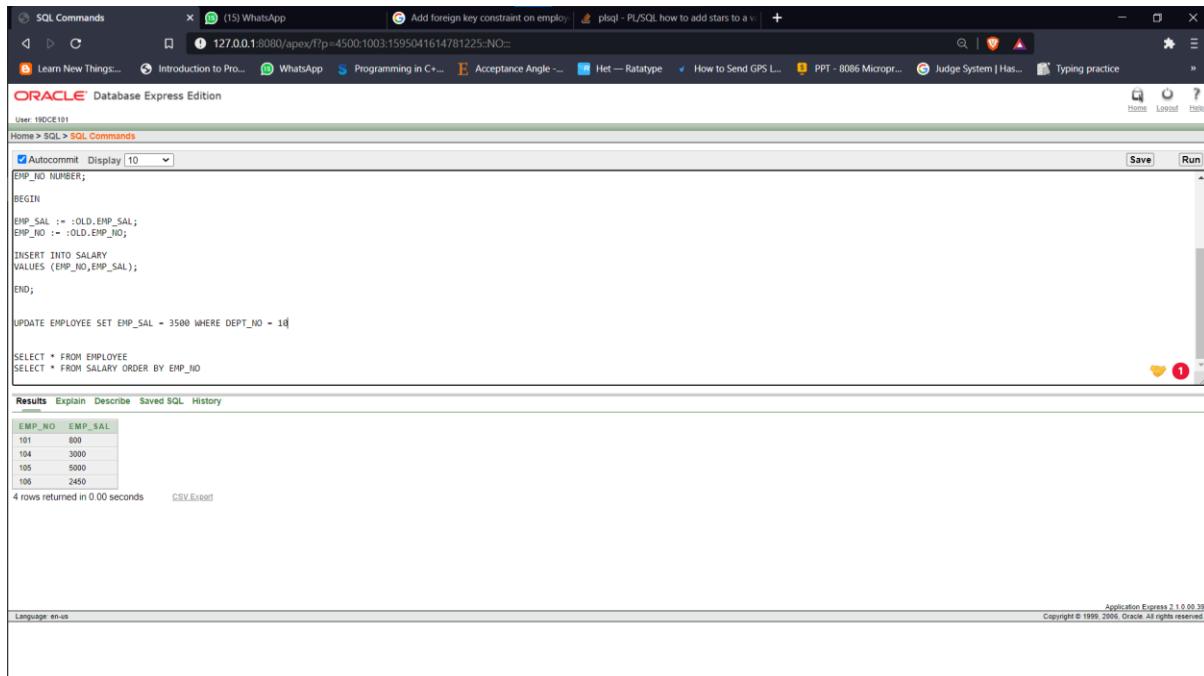
The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, a PL/SQL block is run to update the Employee table. The block uses a trigger-like mechanism to update the salary of employees in department 10 by 3500. A select statement follows to verify the changes.

```

SQL Commands
127.0.0.1:8080/apex/?p=4500:1003:159504161478125:NO:_
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle ... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10 Save Run
EMP_NO NUMBER;
BEGIN
EMP_SAL := :OLD.EMP_SAL;
EMP_NO := :OLD.EMP_NO;
INSERT INTO SALARY
VALUES (EMP_NO,EMP_SAL);
END;
UPDATE EMPLOYEE SET EMP_SAL = 3500 WHERE DEPT_NO = 10
SELECT * FROM EMPLOYEE
SELECT * FROM SALARY ORDER BY EMP_NO
Results Explain Describe Saved SQL History
EMP_NO EMP_NAME EMP_SAL EMP_COMM DEPT_NO L_NAME DEPT_NAME JOB_ID LOCATION MANAGER_ID HIREDATE PHONE NEW_SALARY INCREASE
101 Smith 3500 - 10 shah machine learning it_mgr toronto 105 09-AUG-96 - 920 120
102 Snehal 1600 300 25 gupta data science lec las vegas - 14-MAR-96 - 1840 240
103 RAMESH 1100 500 20 wales machine learning ml_mgr ontario 105 30-NOV-95 - 1265 165
104 Aman 3500 - 10 sharma virtual reality comp_op mexico 12 02-OCT-97 - 3450 450
105 Anita 3500 50000 10 patel big data analytics comp_op germany 107 01-JAN-98 - 5750 750
106 Sneha 3500 24500 10 joseph big data analytics it_acc melbourne 105 28-SEP-97 - 2817.5 367.5
107 Anamika 2975 - 30 jha artificial intelligence it_prog new york - 15-JUL-97 - 342125 446.25
7 rows returned in 0.00 seconds CSV Export
Language: en-us Application Express 2.1.0.09.39
Copyright © 1999, 2006, Oracle. All rights reserved.

```

Salary Table After Command Execution having old values of Emp_Sal:



The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, a PL/SQL block is run to update the Employee table. The block uses a trigger-like mechanism to update the salary of employees in department 10 by 3500. A select statement follows to verify the changes.

```

SQL Commands
127.0.0.1:8080/apex/?p=4500:1003:159504161478125:NO:_
Learn New Things... Introduction to Pro... WhatsApp Programming in C... Acceptance Angle ... Het — Ratatype How to Send GPS L... PPT - 8086 Micropr... Judge System | Has... Typing practice
ORACLE Database Express Edition
User: 19DCE101
Home > SQL > SQL Commands
Autocommit Display 10 Save Run
EMP_NO NUMBER;
BEGIN
EMP_SAL := :OLD.EMP_SAL;
EMP_NO := :OLD.EMP_NO;
INSERT INTO SALARY
VALUES (EMP_NO,EMP_SAL);
END;
UPDATE EMPLOYEE SET EMP_SAL = 3500 WHERE DEPT_NO = 10
SELECT * FROM EMPLOYEE
SELECT * FROM SALARY ORDER BY EMP_NO
Results Explain Describe Saved SQL History
EMP_NO EMP_SAL
101 800
104 3000
105 5000
106 2450
4 rows returned in 0.00 seconds CSV Export
Language: en-us Application Express 2.1.0.09.39
Copyright © 1999, 2006, Oracle. All rights reserved.

```

Conclusion: From this Practical I learned how to use trigger to perform certain tasks.

Practical -18

To solve queries using the concept of View.

- (1) Write a query to create a view for that employee belongs to the location New York.

Code:

--create view

```
CREATE OR REPLACE VIEW NEW_YORK_EMP AS
SELECT * FROM EMPLOYEE WHERE LOCATION = 'new york'
--display view
SELECT * FROM NEW_YORK_EMP;
```

Snap-shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, the following SQL code is entered:

```
CREATE OR REPLACE VIEW NEW_YORK_EMP AS
SELECT * FROM EMPLOYEE WHERE LOCATION = 'new york'
--display view
SELECT * FROM NEW_YORK_EMP;
```

After executing the code, the results are displayed in a table:

| EMP_NO | EMP_NAME | EMP_SAL | EMP_COMM | DEPT_NO | L_NAME | DEPT_NAME | JOB_ID | LOCATION | MANAGER_ID | HIREDATE | PHONE | NEW_SALARY | INCREASE |
|--------|----------|---------|----------|---------|--------|-------------------------|---------|----------|------------|-----------|-------|------------|----------|
| 107 | Anamika | 2975 | - | 30 | jha | artificial intelligence | it_prog | new york | - | 15-JUL-67 | - | 3421.25 | 446.25 |

1 rows returned in 0.07 seconds

- (2) Write a query to create a view for all employee with columns emp_id, emp_name, and job_id.

Code:

```
--create view
CREATE OR REPLACE VIEW EMP1 AS
SELECT EMP_NO,EMP_NAME,JOB_ID FROM EMPLOYEE
--display view
SELECT * FROM EMP1;
```

Snap-shot:

```

CREATE OR REPLACE VIEW EMP1 AS
SELECT * FROM EMPLOYEE
    
```

| EMP_NO | EMP_NAME | JOB_ID |
|--------|----------|----------|
| 101 | Smith | fl_mgr |
| 102 | Snehal | lec |
| 103 | RAMESH | mr_mgr |
| 104 | Aman | comp_oop |
| 105 | Anita | comp_oop |
| 106 | Sneha | fl_acc |
| 107 | Anamika | it_prog |

7 rows returned in 0.00 seconds [CSV Export](#)

(3) Write a query to find the salesmen of the location New York who having salary more than 3000.

Code:

--create view

```

CREATE OR REPLACE VIEW NEW_YORK_EMP AS SELECT * FROM EMPLOYEE
WHERE EMP_SAL > 3000
    
```

--display view

```

SELECT * FROM NEW_YORK_EMP
    
```

Snap-shot

```

SELECT * FROM NEW_YORK_EMP
    
```

| EMP_NO | EMP_NAME | EMP_SAL | DEPT_NAME | DEPT_NO | L_NAME | JOBLD | LOCATION | MANAGER_ID | HIREDATE | PHONE | NEW_SALARY | INCREASE |
|--------|----------|---------|-----------|---------|--------|--------------------|----------|------------|-----------|-------|------------|----------|
| 101 | Smith | 3500 | - | 10 | shah | machine learning | fl_mgr | 105 | 09-AUG-96 | - | 920 | 120 |
| 104 | Aman | 3500 | - | 10 | sharma | virtual reality | comp_oop | 102 | 02-OCT-97 | - | 3450 | 450 |
| 105 | Anita | 3500 | 50000 | 10 | patel | big data analytics | comp_oop | 107 | 01-JAN-98 | - | 5750 | 750 |
| 106 | Sneha | 3500 | 24500 | 10 | joseph | big data analytics | fl_acc | 106 | 26-SEP-07 | - | 2817.5 | 367.5 |

4 rows returned in 0.00 seconds [CSV Export](#)

(4) Write a query to create a view to getting a count of how many employee we have at each department.

Code:

--create view

```
CREATE OR REPLACE VIEW DEPT_COUNTER AS
```

```
SELECT DEPT_NO,COUNT(*)"COUNT" FROM EMPLOYEE GROUP BY DEPT_NO;
```

--display view

```
SELECT * FROM DEPT_COUNTER;
```

Snap-shot:

The screenshot shows a browser window with multiple tabs open. The active tab is titled "SQL Commands". The URL is "127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225::NO=::". The page displays SQL code for creating a view named "DEPT_COUNTER". The code is as follows:

```
CREATE OR REPLACE VIEW DEPT_COUNTER AS
SELECT DEPT_NO,COUNT(*)"COUNT" FROM EMPLOYEE GROUP BY DEPT_NO;
SELECT * FROM DEPT_COUNTER;
```

Below the code, the results of the query are shown in a table:

| DEPT_NO | COUNT |
|---------|-------|
| 25 | 1 |
| 30 | 1 |
| 20 | 1 |
| 10 | 4 |

Text at the bottom of the results area: "4 rows returned in 0.09 seconds" and "CSV Export".

At the bottom right of the page, there is a copyright notice: "Application Express 2.1.0.0.39 Copyright © 1999, 2006, Oracle. All rights reserved."

Conclusion: From This Practical I Learned how to create a view in oracle PL/SQL.

Practical -19

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.

Code:

Using Procedure:

STEP 1: CREATE A PROCEDURE

```
CREATE OR REPLACE PROCEDURE UPDATE_SAL(DEPT_ID IN  
NUMBER,NEW_EMP_SAL IN NUMBER)
```

IS

BEGIN

```
UPDATE EMPLOYEE SET EMP_SAL = NEW_EMP_SAL WHERE DEPT_NO =  
DEPT_ID;
```

```
DBMS_OUTPUT.PUT_LINE('Updated Salary Is: '|| NEW_EMP_SAL);
```

END;

STEP 2: EXECUTE THE COMMAND

```
BEGIN  
UPDATE_SAL(20,1250);  
END;
```

Snap-shot:

```

CREATE OR REPLACE PROCEDURE UPDATE_SAL(DEPT_ID IN NUMBER, NEW_EMP_SAL IN NUMBER)
IS
BEGIN
  UPDATE EMPLOYEE SET EMP_SAL = NEW_EMP_SAL WHERE DEPT_NO = DEPT_ID;
  DBMS_OUTPUT.PUT_LINE('Updated Salary Is: '|| NEW_EMP_SAL);
END;

BEGIN
  UPDATE_SAL(20,1250);
END;

```

Results Explain Describe Saved SQL History

Updated Salary Is: 1250
Statement processed.
0.03 seconds

Language en-us Application Express 7.1.0.0.39
Copyright © 1999, 2006 Oracle. All rights reserved.

Using Function:

STEP 1: CREATE A FUNCTION

```
CREATE OR REPLACE FUNCTION UPDATE_SALARY(DEPT_ID NUMBER,
```

```
EMP_NEW_SAL NUMBER)
```

```
RETURN NUMBER IS
```

```
BEGIN
```

```
  UPDATE EMPLOYEE SET EMP_SAL = EMP_NEW_SAL WHERE DEPT_NO =  
  DEPT_ID;
```

```
  RETURN EMP_NEW_SAL;
```

```
END;
```

STEP 2: USE THAT FUCNTION IN OTHER PL/SQL BLOCKS

```
DECLARE
```

```

NEW_EMP_SAL EMPLOYEE.EMP_SAL%TYPE;
EMP_NO EMPLOYEE.EMP_NO%TYPE;
UPDATED_SAL NUMBER;

BEGIN

NEW_EMP_SAL := 3079;
EMP_NO := 20;
UPDATED_SAL:= 0;
UPDATED_SAL := UPDATE_SALARY(EMP_NO,NEW_EMP_SAL);

DBMS_OUTPUT.PUT_LINE('Updated Salary Is : ' || UPDATED_SAL);

END;

```

Snap-shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, a PL/SQL procedure is written:

```

DECLARE
  NEW_EMP_SAL EMPLOYEE.EMP_SAL%TYPE;
  EMP_NO EMPLOYEE.EMP_NO%TYPE;
  UPDATED_SAL NUMBER;
BEGIN
  NEW_EMP_SAL := 3079;
  EMP_NO := 20;
  UPDATED_SAL:= 0;
  UPDATED_SAL := UPDATE_SALARY(EMP_NO,NEW_EMP_SAL);
  DBMS_OUTPUT.PUT_LINE('Updated Salary Is : ' || UPDATED_SAL);
END;

```

After running the procedure, the results are displayed in the Results tab:

```

Updated Salary Is :3079
1 row(s) updated.
0.11 seconds

```

At the bottom right of the interface, it says "Application Express 2.1.0.00.39" and "Copyright © 1999-2006, Oracle. All rights reserved."

Conclusion: From this Practical I learned how to use Procedure & Functions which are pl/sql objects to perform tasks, making code reusable.

Practical -20

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
    EMPLOYEE_CODE NUMBER;
    AMOUNT NUMBER;
    OPERATION VARCHAR(10);
    invalid_operation EXCEPTION;
    negative_sal EXCEPTION;

BEGIN
    EMPLOYEE_CODE:=EMPLOYEE_CODE;
    AMOUNT:=AMOUNT;
    OPERATION:=OPERATION;

    IF OPERATION = '+' THEN
        UPDATE EMP SET EMP_SAL = EMP_SAL+AMOUNT WHERE EMP_NO =
EMPLOYEE_CODE;
        DBMS_OUTPUT.PUT_LINE('Added '|| AMOUNT || ' to the salary of employee.');
    ELSIF OPERATION = '-' THEN
        UPDATE EMP SET EMP_SAL = EMP_SAL-AMOUNT WHERE EMP_NO =
EMPLOYEE_CODE;
        DBMS_OUTPUT.PUT_LINE('Deducted '|| AMOUNT || ' from the salary of employee.');
        if (AMOUNT < 0) then
            RAISE negative_sal;
        end if;

        else
            RAISE invalid_operation;
    END IF;

EXCEPTION
    when invalid_operation then
        dbms_output.put_line('Invalid Operation !!!');
    when NO_DATA_FOUND then
        dbms_output.put_line('No data Found !!!');

```

```

when negative_sal then
    dbms_output.put_line('Salary cannot be negative !!!');
END;
/

```

Snap-shot:

The screenshot shows the Oracle Database Express Edition interface. In the SQL Commands window, a PL/SQL block is being run. The code handles salary updates based on an operation (+ or -) and includes error handling for negative salaries. The results show that 200 rows were updated, and the execution took 0.00 seconds.

```

SQL Commands
127.0.0.1:8080/apex/f?p=4500:1003:1595041614781225:N:=
ORACLE Database Express Edition
User: TDOCE101
Home > SQL > SQL Commands
Autocommit: Display: 10 Save Run
BEGIN
EMPLOYEE_CODE:=EMPLOYEE_CODE;
AMOUNT:=AMOUNT;
OPERATION:=OPERATION;
IF OPERATION = '+' THEN
    UPDATE EMP SET EMP_SAL = EMP_SAL+AMOUNT WHERE EMP_NO = EMPLOYEE_CODE;
    DBMS_OUTPUT.PUT_LINE('Added'||AMOUNT||' to the salary of employee.');
ELSIF OPERATION = '-' THEN
    UPDATE EMP SET EMP_SAL = EMP_SAL-AMOUNT WHERE EMP_NO = EMPLOYEE_CODE;
    DBMS_OUTPUT.PUT_LINE('Deducted'||AMOUNT||' from the salary of employee.');
IF (AMOUNT < 0) then
    RAISE negative_sal;
end if;
else
    RAISE invalid_operation;
END IF;

```

Results Explain Describe Saved SQL History

Added 200 to the salary of employee.
1 row(s) updated.
0.00 seconds

Language: en-us Application Express 2.1.0.0.39
Copyright © 1999-2006 Oracle All rights reserved.

Conclusion: From This Practical I Learned how to handle exceptions in oracle.

Practical -21

To perform the concept of package

Code:

STEP 1: CREATE A PACKAGE DECLARATION.

```
CREATE OR REPLACE PACKAGE getEmployeeData AS
```

```
-- public
```

```
FUNCTION getRow(Emp_id number) return EMPLOYEE4%ROWTYPE;
```

```
END getEmployeeData;
```

STEP 2: CREATE A PACKAGE BODY.

```
CREATE OR REPLACE PACKAGE BODY getEmployeeData AS
```

```
FUNCTION getRow(Emp_id number) return EMPLOYEE%ROWTYPE AS
```

```
-- THIS VARIABLE WILL BE PRIVATE.
```

```
EMPX EMPLOYEE%ROWTYPE;
```

```
BEGIN
```

```
SELECT * INTO EMPX FROM EMPLOYEE4 WHERE EMP_NO = Emp_id;
```

```
RETURN EMPX;
```

```
END getRow;
```

```
END getEmployeeData;
```

STEP 3: Use the Package In a PL/SQL Block.

```
DECLARE
```

```
EMPX EMPLOYEE%ROWTYPE;
```

```
BEGIN
```

```
EMPX:= getEmployeeData.getRow(101);

DBMS_OUTPUT.PUT_LINE('EMP_SAL: '|| EMPX.EMP_SAL);

-- Similarly all rows and values can be printed.

END;
```

Snap-shot:

The screenshot shows the Oracle Database Express Edition SQL Commands window. The code entered is a PL/SQL package body:

```
BEGIN
SELECT * INTO EMPX FROM EMPLOYEE WHERE EMP_NO = Emp_id;
END getRow;
END getEmployeeData;

DECLARE
EMPX EMPLOYEE%ROWTYPE;
BEGIN
EMPX := getEmployeeData.getRow(101);
DBMS_OUTPUT.PUT_LINE('EMP_SAL: '|| EMPX.EMP_SAL);
-- Similarly all rows and values can be printed.
END;
```

The results pane shows the output of the DBMS_OUTPUT.PUT_LINE statement:

```
EMP_SAL: 3500
Statement processed.
0.00 seconds
```

At the bottom right, there is a small footer: "Application Express 2.1.0.09_39 Copyright © 1999, 2006, Oracle. All rights reserved."

Conclusion: From This Practical I Learned how to create & use a package in oracle.