

Ex. : 02 BASIC SQL QUERIES

Date : 03/01/2019

Aim:

1. Create table DEPT with following columns and data types

Name	Null?	Type

DEPTNO	NOT NULL	NUMBER(2)
DNAME		CHAR(14)
LOCATION		CHAR(13)

Query:

```
SQL> create table DEPT(  
Deptno number(2) not null, Dname char(14), Location char(13));
```

Table created.

2. Insert in the DEPT table the following rows:

DEPTNO	DNAME	LOCATION

10	Accounting	New York
20	Research	Chicago
30	Sales	Dallas
40	Operations	Boston

Query:

```
SQL> insert into DEPT values(&Deptno, '&Dname', '&Location');
```

```
Enter value for deptno: 10
```

```
Enter value for dname: Accounting
```

```
Enter value for location: New York
```

```
old 1: insert into DEPT values(&Deptno, '&Dname', '&Location')
```

```
new 1: insert into DEPT values(10, 'Accounting', 'New York')
```

```
1 row created.
```

```
SQL> insert into DEPT values(&Deptno, '&Dname', '&Location');
```

```
Enter value for deptno: 20
```

```
Enter value for dname: Research
```

```
Enter value for location: Chicago
```

```
old 1: insert into DEPT values(&Deptno, '&Dname', '&Location')
```

```
new 1: insert into DEPT values(20, 'Research', 'Chicago')
```

```
1 row created.
```

```
SQL> insert into DEPT values(&Deptno, '&Dname', '&Location');
```

```
Enter value for deptno: 30
```

```
Enter value for dname: Sales
```

```
Enter value for location: Dallas
```

```
old 1: insert into DEPT values(&Deptno, '&Dname', '&Location')
```

```
new 1: insert into DEPT values(30, 'Sales', 'Dallas')
```

```
1 row created.
```

```
SQL> insert into DEPT values(&Deptno, '&Dname', '&Location');
```

```
Enter value for deptno: 40
```

```
Enter value for dname: Operations
```

```
Enter value for location: Boston
```

```
old 1: insert into DEPT values(&Deptno, '&Dname', '&Location')
```

```
new 1: insert into DEPT values(40, 'Operations', 'Boston')
```

```
1 row created.
```

```
SQL> select * from DEPT;
```

DEPTNO	DNAME	LOCATION
10	Accounting	New York
20	Research	Chicago
30	Sales	Dallas
40	Operations	Boston

```
4 rows selected.
```

3. Add a new department into the DEPT table with a department number of 99, location of Miami, and a department name of Education.

Query:

```
SQL> insert into DEPT values(&Deptno, '&Dname', '&Location');
```

```
Enter value for deptno: 99
```

```
Enter value for dname: Miami
```

```
Enter value for location: Education
```

```
old 1: insert into DEPT values(&Deptno, '&Dname', '&Location')
```

```
new 1: insert into DEPT values(99, 'Education', 'Miami')
```

```
1 row created.
```

```
SQL> select * from DEPT;
```

DEPTNO	DNAME	LOCATION
10	Accounting	New York
20	Research	Chicago
30	Sales	Dallas
40	Operations	Boston
99	Education	Miami

```
5 rows selected.
```

4. Insert yourself as a new employee into EMP table.

Query:

```
SQL> create table emplo(empname varchar(10), salary number(5));
```

Table created.

```
SQL> insert into emplo values('Aashik', 98999);
```

1 row created.

```
SQL> select * from emplo;
```

EMPNAME	SALARY
Aashik	98999

5. Update your own employee data by giving yourself a raise of \$1000 per month.

Query:

```
SQL> update emplo set salary=salary+1000 where empname='Aashik';
```

1 row updated.

6. View the changes you have just done.

Query:

```
SQL> select * from emplo;
```

EMPNAME	SALARY

Aashik	99999

7. Delete yourself from the EMP table.

Query:

```
SQL> delete from emplo where empname='Aashik';
```

1 row deleted.

8. Select from EMP table to check whether the row is existing.

Query:

```
SQL> select * from emplo where empname='Aashik';
```

no rows selected

9. Create a duplicate EMP table: Name is EMPTEST.

Query:

```
SQL> create table emptest as select * from emplo;
```

Table created.

10. Add a new column named SEX to the EMPTEST table with data type of character and length 1.

Query:

```
SQL> alter table emptest add sex char(1);
```

Table altered.

11. Display and see the structure of the table.

Query:

```
SQL> desc emptest;
```

Name	Null?	Type
-----	-----	-----
EMPNAME		VARCHAR2 (10)
SALARY		NUMBER (5)
SEX		CHAR (1)

12. The user have changed their mind. Instead of storing SEX as 'F' or 'M', they want to store SEX as 'MALE' or 'FEMALE'. So, increase the size of the SEX column.

Query:

```
SQL> alter table emp test modify sex char(6);
```

Table altered.

13. Display each employee's name and hiredate.

Query:

```
SQL> alter table emp test add hiredate DATE;
```

Table altered.

```
SQL> insert into emp test  
values('Aashik','99999','Male',date'2019-05-03');
```

1 row created.

```
SQL> select empname, hiredate from emp test;
```

EMPNAME	HIREDATE
Aashik	03-MAY-19

14. Display the information in the above query with hire date appearing first.

Query:

```
SQL> select hiredate, empname from emptest;
```

```
HIREDATE    EMPNAME
```

```
-----
```

```
03-MAY-19  Aashik
```

```
SQL> spool off
```

Ex. : 03 BASIC SQL QUERIES USING OPERATORS

Date : 10/01/2019

Aim:

Create an employee table with the following description:

Name	Type
Emp.No	Number
Name	varchar2
Designation	varchar2
Dept.No	number
Dept.Name	varchar2
Date of joining	Date
Salary	Number

Read the queries and insert 10 valid records accordingly

```
SQL> CREATE TABLE employee(
2  Empno number(5),
3  Name varchar(20),
4  Designation varchar(20),
5  Deptno number(3),
6  Deptname varchar(20),
7  Dateofjoin date,
8  salary number
9  );
```

Table created.

```
SQL> desc employee
```

Name	Null?	Type
EMPNO		NUMBER (5)
NAME		VARCHAR2 (20)
DESIGNATION		VARCHAR2 (20)
DEPTNO		NUMBER (3)
DEPTNAME		VARCHAR2 (20)
DATEOFJOIN		DATE
SALARY		NUMBER

```
SQL> select * from employee;
```

EMPNO	NAME	DESIGNATION	DEPTNO	DEPTNAME	DATEOFJOI	SALARY
-----	-----	-----	-----	-----	-----	-----
1746	keerthi	inspector			28-MAY-91	6000
7369	ishwarya	manager	10	sales	29-SEP-80	5000
7521	dd	manager	20	food	25-SEP-80	5000
7581	divya	salesman	10	sales	25-SEP-81	1000
7647	abc	clerk	30	admin	25-SEP-76	500
5678	Aashiki	analyst	40	it	12-JUL-97	3000
5356	harsha	manager	40	it	02-MAR-83	7000
1234	abi	clerk	20	food	01-JAN-90	1000
7646	das	accountant	50	accounts	30-MAR-71	8000
2344	shwetha	helper	20	food	25-NOV-82	1000
6876	srinivas	clerk	50	accountant	05-AUG-57	2000

1) List all the employees belonging to department number 10.

```
SQL> SELECT Name from employee
2   where Deptno='10';
NAME
-----
ishwarya
divya
```

2) List the name and salary of the employees whose salary is greater than 1000.

```
SQL> select name,salary from employee
2   where Salary>1000;
```

NAME	SALARY
-----	-----
ishwarya	5000
dd	5000
Aashiki	3000
harsha	7000
das	8000
srinivas	2000

6 rows selected.

3) List the name of the clerks working in department number 20.

```
SQL> select Name from employee
2   where Designation='clerk' and Deptno='20';
```

```
NAME
-----
Abi
```

4) List the employee number and name of the managers.

```
SQL> select Empno,Name from employee
      2  where Designation='manager';
```

```
EMPNO NAME
-----
7369  ishwarya
7521  dd
5356  harsha
```

5) List the names of analysts and salesmen.

```
SQL> select Name from employee
      2  where Designation='analayst'or Designation='salesman';
```

```
NAME
-----
Divya
Aashiki
```

6) List the details of the employees who have joined before the end of 30 September 80.

```
SQL> select name from employee
      2  where Dateofjoin<date'1980-09-30';
```

```
NAME
-----
ishwarya
dd
abc
das
srinivas
```

7) List the employees who are not managers.

```
SQL> select Name from employee
      2  where Designation!='manager';
```

```
NAME
-----
divya
abc
Aashiki
abi
das
shwetha
srinivas
```

7 rows selected.

8) List the name of the employees whose employee numbers are 7369 and 7521.

```
SQL> select Name from employee
      2  where Empno='7369' or Empno='7521';
```

```
NAME
-----
ishwarya
dd
```

9) List the details of the employees not belonging to the departments 10, 30 and 40.

```
SQL> select Name from employee
      2  where Deptno!=10 and Deptno!=30 and Deptno!=40;
```

```
NAME
-----
dd
abi
das
shwetha
srinivas
```

10) List the name and salary for the employees whose salary is between 1000 and 2000.

```
SQL> select Name,Salary from employee
      2  where Salary between 1000 and 2000;
```

NAME	SALARY
-----	-----
divya	1000
abi	1000
shwetha	1000
srinivas	2000

11) List the names of the employees who have joined before 30 June 81 and after 31 Dec 81.

```
SQL> select Name from employee
      2  where Dateofjoin<date'1981-06-30' or
      3  Dateofjoin>date'1981-12-31';
```

```
NAME
-----
ishwarya
dd
abc
Aashiki
harsha
abi
das
```

```
shwetha
srinivas
```

9 rows selected.

12) List the different jobs in the employee table.

```
SQL> select distinct Designation from employee;
```

```
DESIGNATION
-----
salesman
helper
clerk
accountant
manager
analyst
```

6 rows selected.

13) List the names of the employees who are not eligible for commission.

```
SQL> select Name from employee
  2  where Designation NOT IN
  ('manager','salesman','analyst','accountant','inspector');
```

```
NAME
-----
abc
abi
shwetha
srinivas
```

14) List the name and designation of the employees who does not report to anybody.

```
SQL> select Name,Designation from employee where
  2  Deptno IN('30','50','40');
```

NAME	DESIGNATION
-----	-----
abc	clerk
Aashiki	analyst
harsha	manager
das	accountant
srinivas	clerk

15) List the details of employees who are not assigned to any department.

```
SQL> select Empno,Name,Designation,Dateofjoin,Salary from
      employee where
      2 Deptno IS NULL and Deptname IS NULL;
```

EMPNO	NAME	DESIGNATION	DATEOFJOI	SALARY
-----	-----	-----	-----	-----
1746	keerthi	inspector	28-MAY-91	6000

16) List the details of employees who are eligible for commission.

```
SQL> select * from employee
      2 where Designation
      IN('manager','salesman','analyst','accountant','inspector')
      ;
```

EMPNO	NAME	DESIGNATION	DEPTNO	DEPTNAME	DATEOFJOI	SALARY
-----	-----	-----	-----	-----	-----	-----
1746	keerthi	inspector			28-MAY-91	6000
7369	ishwarya	manager	10	sales	29-SEP-80	5000
7521	dd	manager	20	food	25-SEP-80	5000
7581	divya	salesman	10	sales	25-SEP-81	1000
5678	Aashiki	analyst	40	it	12-JUL-97	3000
5356	harsha	manager	40	it	02-MAR-83	7000
7646	das	accountant	50	accounts	30-MAR-71	8000

7 rows selected.

17) List the details of employees whose salary is greater than 2000 and have no commission.

```
SQL> select * from employee
      2 where Salary>=2000 and Designation IN ('clerk',
      'helper');
```

EMPNO	NAME	DESIGNATION	DEPTNO	DEPTNAME	DATEOFJOI	SALARY
-----	-----	-----	-----	-----	-----	-----
6876	srinivas	clerk	50	accounts	05-AUG-57	2000

18) List the details of employees whose name start with 'S'.

```
SQL> select * from employee
      2 where Name LIKE 's%';
```

EMPNO	NAME	DESIGNATION	DEPTNO	DEPTNAME	DATEOFJOI	SALARY
-----	-----	-----	-----	-----	-----	-----
2344	shwetha	helper	20	food	25-NOV-82	1000
6876	srinivas	clerk	50	accountant	05-AUG-57	2000

19) List the names of employees whose name end with 's'.

```
SQL> select * from employee
      2  where Name LIKE '%s';
```

EMPNO	NAME	DESIGNATION	DEPTNO	DEPTNAME	DATEOFJOI	SALARY
-----	-----	-----	-----	-----	-----	-----
6876	srinivas	clerk	50	accounts	05-AUG-57	2000

20) List the names of the employees whose name has exactly 5 characters.

```
SQL> select Name from employee
      2  where LENGTH(Name)=5;
```

```
NAME
-----
divya
```

21) List the names of employees whose name has 'i' as second character.

```
SQL> select Name from employee
      2  where Name LIKE '_i%';
```

```
NAME
-----
divya
```

22) List the names of the employees in ascending order of their salaries.

```
SQL> select Name from employee
      2  ORDER BY Salary;
```

```
NAME
-----
abc
shwetha
abi
divya
srinivas
Aashiki
ishwarya
dd
keerthi
harsha
das
```

```
11 rows selected.
```


23) List the name, salary and PF amount which is 10% of the salary of all employees.

```
SQL> select Name,Salary,Salary*0.10 PF from employee;
```

NAME	SALARY	PF
keerthi	6000	600
ishwarya	5000	500
dd	5000	500
divya	1000	100
abc	500	50
Aashiki	3000	300
harsha	7000	700
abi	1000	100
das	8000	800
shwetha	1000	100
srinivas	2000	200

11 rows selected.

24) List the name, salary, designation and department number of employees in the descending order of their department number.

```
SQL> select Name,Salary,Designation,Deptno from employee
2 order by Deptno desc;
```

NAME	SALARY	DESIGNATION	DEPTNO
keerthi	6000	inspector	
das	8000	accountant	50
srinivas	2000	clerk	50
Aashiki	3000	analyst	40
harsha	7000	manager	40
abc	500	clerk	30
dd	5000	manager	20
shwetha	1000	helper	20
abi	1000	clerk	20
ishwarya	5000	manager	10
divya	1000	salesman	10

11 rows selected.

25) List the total number of employees.

```
SQL> select count(*) from employee;
COUNT(*)
```

```
-----
11
```

26) List the number of distinct jobs available.

```
SQL> select count(distinct Designation) from employee;
```

```
COUNT (DISTINCTDESIGNATION)
-----
                                7
```

27) List the total salaries of the employees.

```
SQL> select sum(Salary) AS "Total Salary" from employee;
```

```
Total Salary
-----
          39500
```

28) List the average salary and number of employees belonging to department number 20.

```
SQL> select avg(Salary),count(Empno) from employee
      2  where Deptno='20';
```

```
AVG (SALARY)  COUNT (EMPNO)
-----
2333.33333          3
```

29) List the department number and number of employees in each department.

```
SQL> select Deptno,count(*) from employee group by Deptno;
```

```
DEPTNO      COUNT (*)
-----
          1
          30          1
          20          3
          40          2
          50          2
          10          2
```

6 rows selected.

30) List the department number and total of salaries in each department.

```
SQL> select Deptno,sum(Salary) from employee group by
      Deptno;
```

```
DEPTNO      SUM (SALARY)
-----
          6000
          30          500
          20          7000
          40         10000
```

50	10000
10	6000

6 rows selected.

31) List the jobs and number of employees in each job. The result should be in ascending order of number of employees in each department

```
SQL> select Designation,count(Designation) from employee
group by Designation order by count(Designation);
```

DESIGNATION	COUNT (DESIGNATION)
salesman	1
helper	1
inspector	1
analyst	1
accountant	1
manager	3
clerk	3

7 rows selected.

32) List the jobs, maximum salary, minimum salary, average salary from employees according to their designation.

```
SQL> select Designation,max(Salary),min(Salary),avg(Salary)
from employee group by Designation;
```

DESIGNATION	MAX (SALARY)	MIN (SALARY)	AVG (SALARY)
salesman	1000	1000	1000
helper	1000	1000	1000
inspector	6000	6000	6000
clerk	2000	500	1166.66667
accountant	8000	8000	8000
manager	7000	5000	5666.66667
analyst	3000	3000	3000

7 rows selected.

33) List the average salary of employees for each job excluding the manager.

```
SQL> select Designation, avg(Salary) from employee where
Designation <> 'manager'
2 group by Designation;
```

DESIGNATION	AVG (SALARY)
salesman	1000
helper	1000
inspector	6000
clerk	1166.66667
accountant	8000
analyst	3000

6 rows selected.

34) List the department number, job and average salary of employees for all jobs according to their department numbers.

```
SQL> select Deptno, Designation, avg(Salary) from employee
group by Designation, Deptno;
```

DEPTNO	DESIGNATION	AVG (SALARY)
50	accountant	8000
	inspector	6000
20	helper	1000
10	manager	5000
40	analyst	3000
20	manager	5000
10	salesman	1000
20	clerk	1000
40	manager	7000
50	clerk	2000
30	clerk	500

11 rows selected.

35) List the department number and average salary of employees belonging to departments which employ more than 5 employees.

```
SQL> select deptno, avg(Salary) from employee
2 group by deptno
3 having count(Empno) > 5;
```

no rows selected

36) List the jobs of employees where the maximum salary of the job is greater than or equal to 5000.

```
SQL> select Designation from employee
      2  group by Designation having max(Salary)>=5000;
```

```
DESIGNATION
-----
inspector
accountant
manager
```

37) List the total salary, minimum salary, maximum salary and average salary of the employees belonging to department 20. Display only those rows having their average salary greater than 1000.

```
SQL> select sum(Salary) AS Total_Salary, min(Salary) AS
Min_Salary, max(Salary) AS Max_Salary, avg(Salary) AS
Avg_Salary from employee
      2  where Deptno=20
      3  having avg(Salary)>1000;
```

```
TOTAL_SALARY MIN_SALARY MAX_SALARY AVG_SALARY
-----
          7000          1000          5000 2333.33333
```

Ex. : 04 SQL QUERIES USING BUILT-IN FUNCTIONS

Date : 17/01/2019

Aim:

- 1. Write a query using built-in function to round-off a number to nearest whole number.**

```
SQL> select round(25.45) as Round_off from dual;
      ROUND_OFF
-----
          25
```

- 2. Write a query using built-in function to find the absolute value of a number.**

```
SQL> select abs(-34.67) as Absolute_value from dual;

      ABSOLUTE_VALUE
-----
              34
```

- 3. Write a query using built-in function to retain the decimal part and truncate the fractional part in a number.**

```
SQL> select floor(678.124) as Decimal_Part from dual;

      DECIMAL_PART
-----
             678
```

- 4. Write a query using built-in function to find the remainder as a result of division between two numbers.**

```
SQL> select mod(6,4) as Remainder from dual;

      REMAINDER
-----
              2
```

- 5. Write a query using built-in function to find the sign of a value without its magnitude**

```
SQL> select sign(-35.16) as Sign from dual;
      SIGN
-----
       -1
```

```
SQL> select sign(0) as Sign from dual;
```

```
      SIGN
-----
        0
```

```
SQL> select sign(346) as Sign from dual;
```

```
      SIGN
-----
        1
```

- 6. Write a query using built-in function to find the value of a number m raised to power n.**

```
SQL> select power(6,3) as Power from dual;
```

```
      POWER
-----
       216
```

- 7. Write a query using built-in function to find the square root of a number.**

```
SQL> select sqrt(3846) as Square_root from dual;
```

```
 SQUARE_ROOT
-----
  62.0161269
```

- 8. Write a query using built-in function to round off a number to the nearest integer with respect to the decimal places.**

```
SQL> select ceil(45.836) as Nearest_integer from dual;
```

```
NEAREST_INTEGER
-----
              46
```

- 9. Write a query using built-in function to find the exponential of a number.**

```
SQL> select exp(3) as Exponential from dual;
```

```
EXPONENTIAL
-----
  20.0855369
```

- 10. Write a query using built-in function to accept a character/ column of characters as input and return as output the initial character in uppercase.**

```
SQL> select INITCAP('hello i am fine') as INITIAL_CAP from dual;
```

```
INITIAL_CAP
-----
Hello I Am Fine
```

11. Write a query using built-in function to accept a character/ column of characters as input and return as output in uppercase and lowercase.

```
SQL> select UPPER('hello i am fine') as UPPER from dual;
```

```
UPPER
-----
HELLO I AM FINE
```

```
SQL> select LOWER('HELLO I Am Fine') as LOWER from dual;
```

```
LOWER
-----
hello i am fine
```

12. Write a query using built-in function to concatenate two strings.

```
SQL> select concat('data','base') as CONCATENATION from dual;
```

```
CONCATEN
-----
database
```

13. Write a query using built-in function to find character equivalent of a number.

```
SQL> select CHR(35),chr(97) from dual;
```

```
C C
- -
# a
```

14. Write a query using built-in function to left pad and right pad a string with *.

```
SQL> select rpad('hello',10,'*') from dual;
```

```
RPAD('HELL
-----
hello*****
```

```
SQL> select lpad('hello',10,'*') from dual;
```

```
LPAD('HELL
-----
*****hello
```


15. Write a query using built-in function to perform right and left trim of a string.

```
SQL> select ltrim('      Hello') from dual;
```

```
LTRIM
```

```
-----
```

```
Hello
```

```
SQL> select rtrim('Hello      ') from dual;
```

```
RTRIM
```

```
-----
```

```
Hello
```

```
SQL> select trim('      Hello      ') from dual;
```

```
TRIM(
```

```
-----
```

```
Hello
```

16. Write a query using built-in function to extract specified number of characters from a string.

```
SQL> select substr('Welcome',3,4) from dual;
```

```
SUBS
```

```
----
```

```
lcom
```

17. Write a query using built-in function to return the length of a string up to the first occurrence of the character specified.

```
SQL> select INSTR('ishwarya','a') from dual;
```

```
INSTR(' ISHWARYA', 'A')
```

```
-----
```

```
5
```

18. Write a query using built-in function to find the ascii equivalent of a character given as input.

```
SQL> select ascii('C') from dual;
```

```
ASCII('C')
```

```
-----
```

```
67
```

19. Write a query using built-in function to find the length of a string/ group of string given as input.

```
SQL> select length('This is database using oracle sql!')
from dual;
```

```
LENGTH('THISISDATABASEUSINGORACLESQL!')
-----
```

34

- 20. Write a query using built-in function to extract specified number of characters and replace with new characters from specified string.**

```
SQL> select replace('harshini','har','var') from dual;
```

```
REPLACE (
-----
varshini
```

- 21. Write a query using built-in function to replace a particular character in a string by a particular character.**

```
SQL> select replace('harshini','h','v') from dual;
```

```
REPLACE (
-----
varsvini
```

- 22. Write a query using built-in function to find today's date.**

```
SQL> select sysdate from dual;
```

```
SYSDATE
-----
17-JAN-19
```

- 23. Write a query using built-in function to increase a particular/ group of date by number of months specified.**

```
SQL> select add_months(sysdate,9) from dual;
```

```
ADD_MONTH
-----
17-OCT-19
```

- 24. Write a query using built-in function to find the last date of the month in the particular date specified.**

```
SQL> select last_day(date'1978-04-07') from dual;
```

```
LAST_DAY (
-----
30-APR-78
```

25. Write a query using built-in function to find the number of months between two date.

```
SQL> select months_between(date'2013-04-30',date'2013-01-31') from dual;
```

```
MONTHS_BETWEEN (DATE'2013-04-30',DATE'2013-01-31')
```

```
-----  
3
```

26. Write a query using built-in function to round off a particular date to the next month that follows.

```
SQL> select round(sysdate,'month') from dual;
```

```
ROUND (SYS
```

```
-----
```

```
01-FEB-19
```

27. Write a query using built-in function to round off the date to the nearest Sunday.

```
SQL> select trunc(sysdate,'DAY') from dual;
```

```
TRUNC (SYS
```

```
-----
```

```
13-JAN-19
```

Ex. : 05 SQL QUERIES USING SET OPERATIONS, JOIN
Date : 10/01/2019 & SUBQUERIES

Aim:

Create the following tables :

Bus-details

```
SQL>
1 CREATE TABLE Bus_details(
2 Bus_code number primary key,Bus_desc varchar(20));
/
SQL> select * from bus_details;
```

BUS_CODE	BUS_DESC
100	delux
200	super_fast
300	AC

Bus-route

```
SQL>
1 CREATE TABLE Bus_route(
2 Route_id number primary key,Route_no number unique,
3 Bus_code number,Origin varchar(20),destination varchar(20),
4 fare number,dist number,capacity number,
5 FOREIGN KEY(Bus_code) REFERENCES Bus_details(Bus_code));
Table created.
SQL> select * from bus_route;
```

ROUTE_ID	ROUTE_NO	BUS_CODE	ORIGIN	DESTINATION	FARE	DIST	CAPACITY
11	550	100	Chennai	Trichy	700	350	50
2120	570	200	Bangalore	Chennai	800	400	45
13	882	300	Delhi	Chennai	1000	2000	60
14	750	100	Chennai	Pondicherry	400	200	40
15	670	200	Mumbai	Kolkatta	900	1800	42

Journey

```
SQL>
1 CREATE TABLE Journey(
2 Journey_id number primary key, dateofjourney date NOT NULL,
3 Time varchar(20) NOT NULL,Route_id number,
```

```

4  Bus_code number,
5  FOREIGN KEY(Route_id) REFERENCES Bus_route(Route_id),
6  FOREIGN KEY(Bus_code) REFERENCES Bus_details(Bus_code));

```

Table created.

```
SQL> select * from journey;
```

JOURNEY_ID	DATEOFJOU	TIME	ROUTE_ID	BUS_CODE
2	31-DEC-18	23:00	2120	200
10	26-JAN-18	20:00	15	200
5	30-OCT-18	22:00	13	300
20	28-MAY-19	21:00	11	100
30	01-JUL-18	10:00	14	100
25	15-SEP-18	18:00	2120	200
1	17-JAN-19	10:00	14	100

Ticket

```

SQL> CREATE TABLE Ticket(
2  Journey_id number, Ticket_no number primary key,
3  Dateofbirth date, dateofjourney date,
4  Time varchar(20) NOT NULL, Station varchar(20),
5  Origin varchar(20) NOT NULL,
6  destination varchar(20) NOT NULL, Adults number,
7  Total_fare number, Route_id number,
8  FOREIGN KEY(Journey_id) REFERENCES Journey(Journey_id),
9  FOREIGN KEY(Route_id) REFERENCES Bus_route(Route_id));

```

Table created.

J_ID	T_NO	DAOFB	DAOJ	TIME	STAT	ORGN	DEST	ADTS	TOT_F	R_ID
25	2105	29JAN97	15SEP18	18:00	Bglr	Bangalr	Chennai	2	1600	2120
2	1200	09FEB00	31DEC18	23:00	Bglr	Bangalr	Chennai	1	800	2120
20	2210	28FEB00	28MAY19	21:00	Tbm	Chennai	Trichy	3	2100	11
5	1203	21NOV72	30OCT18	22:00	Delhi	Delhi	Chennai	2	2000	13
30	3208	04APR99	01JUL18	10:00	Tvmyr	Chennai	Pondy	4	1600	14
1	2202	10OCT65	17JAN19	10:00	Klpkm	Chennai	Pondy	3	1200	14
10	1250	18JAN90	26JAN18	20:00	Mumbai	Mumbai	Klkta	2	1800	15

Ticket_detail

```

SQL> CREATE TABLE Ticket_detail(
2  Ticket_no number, Name varchar(20), Sex char(10),
3  Age number, Fare number);

```

Table created.

```
SQL> select * from ticket_detail;
```

TICKET_NO	NAME	SEX	AGE	FARE
2105	John	M	22	1600
1200	Anne	F	19	800
2210	Ramya	F	19	2100
1203	Patrick	M	47	2000
3208	Ram	M	20	1600
2202	Sam	M	54	1200
1250	Raj	M	29	1800

1. Display the bus description which is having the least capacity.

```
SQL> select b1.bus_desc,b2.capacity from bus_details b1 join
bus_route b2 on b1.bus_code=b2.bus_code and b2.capacity=(select
min(capacity) from bus_route);
```

BUS_DESC	CAPACITY
delux	40

2. How many buses are having destination as Chennai?

```
SQL> select count(*) from bus_route where destination='Chennai';
```

COUNT (*)
2

3. How many passengers are traveling below 21 years of age?

```
SQL> select count(age) from ticket_detail where age<21;
```

COUNT (AGE)
3

4. Display the bus description which is having the highest fare.

```
SQL> select b1.bus_desc,b2.fare from bus_details b1 join
bus_route b2 on b1.bus_code=b2.bus_code and b2.fare=(select
max(fare) from bus_route);
```

BUS_DESC	FARE
AC	1000

5. Display the names of the passengers who have booked their ticket in the month of January.

```
SQL> select t1.name from ticket_detail t1 join ticket t2 on
t1.ticket_no=t2.ticket_no and extract(month from
t2.dateofjourney)=1;
```

NAME

Raj

Sam

6. Display the description and Bus Code of the bus whose fare is greater than the average fare in the table.

```
SQL> select distinct b1.bus_desc,b1.bus_code from bus_details b1
join bus_route b2 on b1.bus_code=b2.bus_code and b2.fare>(select
avg(fare) from bus_route);
```

BUS_DESC BUS_CODE

super_fast 200

AC 300

7. How many female passengers are traveling in the Deluxe Bus?

```
SQL> select count(sex) from ticket_detail t1 join ticket t2 on
t1.ticket_no=t2.ticket_no and t2.route_id IN(select route_id
from ticket where route_id='11' or route_id='14') and t1.sex
IN(select sex from ticket_detail where sex='F');
```

COUNT (SEX)

1

8. How many male passengers are traveling in the Super Fast Bus?

```
SQL> select count(sex) from ticket_detail t1 join ticket t2 on
t1.ticket_no=t2.ticket_no and t2.route_id IN(select route_id
from ticket where route_id='2120' or route_id='15') and t1.sex
IN(select sex from ticket_detail where sex='M');
```

COUNT (SEX)

2

9. Display the names of the passengers who departure from Bangalore.

```
SQL> select t1.name from ticket_detail t1 join ticket t2 on
t1.ticket_no=t2.ticket_no and t2.origin IN(select origin from
ticket where origin='Bangalore');
```

NAME

John

Anne

10. Display the journey time of the passenger "John".

```
SQL> select t1.name,t2.time from ticket_detail t1 join ticket t2
on t1.ticket_no=t2.ticket_no and t1.name=(select name from
ticket_detail where name='John');
```

NAME

TIME

John

18:00

11. Display the bus description which are neither originating from Chennai nor reaching Chennai.

```
SQL> select b1.bus_desc from bus_details b1 join bus_route b2 on
b1.bus_code=b2.bus_code and NOT(b2.origin='Chennai' or
b2.destination='Chennai');
```

BUS_DESC

super_fast

12. Select rows from Bus route such that the route id's are greater than any of the ticket nos with J_id as 02 in the journey table.

```
SQL> select * from bus_route
2 where route_id>(select t1.ticket_no from ticket t1 inner
join journey j on t1.journey_id=j.journey_id where
j.journey_id='2');
```

R_ID	R_NO	BUS_CODE	ORIGIN	DESTINATION	FARE	DIST	CAPACITY
----	----	-----	-----	-----	-----	-----	-----
2120	570	200	Bangalore	Chennai	800	400	45

13. Select rows from Bus route such that the route id's are greater than all the ticket nos with J_id as 02 in the journey table.

```
SQL> select * from bus_route where route_id>(select
max(ticket_no) from ticket inner join journey on
ticket.journey_id=journey.journey_id where
journey.journey_id='2');
```

ROUTE_ID	R_NO	BUS_CODE	ORIGIN	DESTINATION	FARE	DIST	CAPACITY
-----	-----	-----	-----	-----	-----	-----	-----
2120	570	200	Bangalore	Chennai	800	400	45

14. Select rows from ticket such that the ticket number exceeds the average of the total fare and the origin for such number should be Chennai.

J_ID	T_NO	DOB	DOFJ	TIME	STAT	ORIGIN	DEST	ADLTS	TO_F	R_ID
1	2202	10OCT65	17JAN19	10:00	Klpkm	Chn	Pondy	3	1200	14
20	2210	28FEB00	28MAY19	21:00	Tbm	Chn	Trichy	3	2100	11
30	3208	04APR99	01JUL18	10:00	Trvnmr	Chn	Pondy	4	1600	14

15. Select distinct route id's from bus route and ticket tables.

```
SQL> select route_id from bus_route union select route_id from
ticket;
```

ROUTE_ID
11
13
14
15
2120

Ex. : 06 SQL VIEWS, INDEX, SEQUENCES & SYNONYMS
Date : 31/01/2019

Aim:

1. Create a view jview from journey table such that it contains day, time and route id with j_day, j_time and j_rid as column headings.

```
SQL> create view j_view(j_day,j_time,j_rid) as select
Dateofjourney,Time,Route_id from journey;
View created.
```

```
SQL> select * from j_view;
```

J_DAY	J_TIME	J_RID
31-DEC-18	23:00	2120
26-JAN-18	20:00	15
30-OCT-18	22:00	13
28-MAY-19	21:00	11
01-JUL-18	10:00	14
15-SEP-18	18:00	2120
17-JAN-19	10:00	14

2. Update the jview such that j_day is 20-jan-98 and j_rid is 301.

```
SQL> update j_view set j_day = date'1998-01-20' where
j_rid='15';
1 row updated.
```

```
SQL> select * from j_view;
```

J_DAY	J_TIME	J_RID
31-DEC-18	23:00	2120
20-JAN-98	20:00	15
30-OCT-18	22:00	13
28-MAY-19	21:00	11
01-JUL-18	10:00	14
15-SEP-18	18:00	2120
17-JAN-19	10:00	14

7 rows selected.

3. Select the contents of the corresponding table that jview is based and check whether the update has occurred.

```
SQL> select * from journey;
```

JOURNEY_ID	DATEOFJOU	TIME	ROUTE_ID	BUS_CODE
2	31-DEC-18	23:00	2120	200
10	20-JAN-98	20:00	15	200
5	30-OCT-18	22:00	13	300
20	28-MAY-19	21:00	11	100
30	01-JUL-18	10:00	14	100
25	15-SEP-18	18:00	2120	200
1	17-JAN-19	10:00	14	100

7 rows selected.

4. Create a synonym passenger for ticketdetail table.

```
SQL> create synonym passenger for ticket_detail;
```

Synonym created.

5. Display the contents of passenger.

```
SQL> select * from passenger;
```

TICKET_NO	NAME	SEX	AGE	FARE
2105	John	M	22	1600
1200	Anne	F	19	800
2210	Ramya	F	19	2100
1203	Patrick	M	47	2000
3208	Ram	M	20	1600
2202	Sam	M	54	1200
1250	Raj	M	29	1800

7 rows selected.

6. Create a synonym bus details for busroute1 table.

```
SQL> create synonym busdetails for bus_route;
```

Synonym created.

```
SQL> select * from busdetails;
```

R_ID	R_NO	BUS_CODE	ORIGIN	DESTINATION	FARE	DIST	CAPACITY
------	------	----------	--------	-------------	------	------	----------

```

-----
11    550    100      Chennai  Trichy      700    350    50
2120  570    200      Bangalore Chennai    800    400    45
13    882    300      Delhi    Chennai    1000   2000    60
14    750    100      Chennai  Pondicherry 400    200    40
15    670    200      Mumbai   Kolkatta    900   1800    42

```

7. Drop the passenger synonym created previously.

```
SQL> drop synonym passenger;
```

Synonym dropped.

8. Create an index on route_id column on busroute table.

```
SQL> create index r_id on bus_route(route_id);
create index r_id on bus_route(route_id)
                                *
```

ERROR at line 1:

ORA-01408: such column list already indexed

```
SQL> create index routeindex on bus_route(dist);
```

Index created.

9. Drop the index what you had created.

```
SQL> drop index routeindex;
```

Index dropped.

10. Create a sequence ticket where minimum value is 1 and maximum value is 20 and increment it with 2 starting with 1.

```
SQL> create sequence ticket1 START WITH 1 INCREMENT BY 2
MINVALUE 1 MAXVALUE 20;
```

Sequence created.

11. Insert the sequence ticket into the ticket column of ticket table.

```
SQL> insert into ticket(ticket_no,time,origin,destination)
values(ticket1.nextval,'12:00','Chennai','Madurai');
```

1 row created.

```
SQL> select * from ticket;
```

J_ID	T_NO	DATEOB	DATEOJ	TIME	STAT	ORIG	DESTI	ADTS	TOT_F	R_ID
25	2105	29JAN97	15SEP18	18:00	Bglre	Bglre	Chn	2	1600	2120
2	1200	09FEB00	31DEC18	23:00	Bglre	Bglre	Chn	1	800	2120
20	2210	28FEB00	28MAY19	21:00	Tbm	Chn	Trichy	3	2100	11
5	1203	21NOV72	30OCT18	22:00	Delhi	Delhi	Chnn	2	2000	13
30	3208	04APR99	01JUL18	10:00	Trnmr	Chnn	Pondy	4	1600	14
1	2202	10OCT65	17JAN19	10:00	Klpkm	Chnn	Pondy	3	1200	14
10	1250	18JAN90	26JAN18	20:00	Mumb	Mumb	Klkta	2	1800	15
1				12:00	Chn	Madurai				

12. Alter the sequence such that the maximum vale is 25.

```
SQL> alter sequence ticket1 MAXVALUE 25;
```

Sequence altered.

13. List all the sequences created by you.

```
SQL> select sequence_name from user_sequences;
```

SEQUENCE_NAME

DEPARTMENTS_SEQ

EMPLOYEES_SEQ

LOCATIONS_SEQ

TICKET1

14. List all the views created by you.

```
SQL> select view_name from user_views;
```

VIEW_NAME

EMP_DETAILS_VIEW

JVIEW

J_VIEW

15. List all the indexes created by you.

```
SQL> select index_name from user_indexes;
```

INDEX_NAME

SYS_C0011609

SYS_C0011603
ROUTEINDEX
SYS_C0011598
SYS_C0011599
SYS_C0011597
JHIST_EMP_ID_ST_DATE_PK
JHIST_JOB_IX
JHIST_EMPLOYEE_IX
JHIST_DEPARTMENT_IX
EMP_EMAIL_UK
EMP_EMP_ID_PK
EMP_DEPARTMENT_IX
EMP_JOB_IX
EMP_MANAGER_IX
EMP_NAME_IX
JOB_ID_PK
DEPT_ID_PK
DEPT_LOCATION_IX
LOC_ID_PK
LOC_CITY_IX
LOC_STATE_PROVINCE_IX
LOC_COUNTRY_IX
COUNTRY_C_ID_PK
REG_ID_PK

16. Drop all the database objects created by you.

SQL> drop index routeindex;

Index dropped.

SQL> drop view jview;

View dropped.

SQL> drop view j_view;

View dropped.

SQL> drop sequence ticket1;

Sequence dropped.

Ex. : 07 PROCEDURES & FUNCTIONS

Date : 14/02/2019

Aim:

1. Write a PL/SQL procedure to print the Armstrong number.

```
1  declare
2      n number:=&n;
3      s number:=0;
4      r number;
5      len number;
6      m number;
7  begin
8      m:=n;
9      len:=length(to_char(n));
10     while n>0
11     loop
12         r:=mod(n,10);
13         s:=s+power(r,len);
14         n:=trunc(n/10);
15     end loop;
16     if m=s
17     then
18         dbms_output.put_line('armstrong number');
19     else
20         dbms_output.put_line('not armstrong number');
21     end if;
22* end;
23  /
```

Enter value for n: 153
old 2: n number:=&n;
new 2: n number:=153;
armstrong number

PL/SQL procedure successfully completed.

2. Write a PL/SQL code to reverse a given number.

```
SQL> declare
  2  n number;
  3  i number;
  4  rev number:=0;
  5  r number;
  6
  7  begin
  8  n:=&n;
  9
 10  while n>0
 11  loop
 12  r:=mod(n,10);
 13  rev:=(rev*10)+r;
 14  n:=trunc(n/10);
 15  end loop;
 16
 17  dbms_output.put_line('reverse is '||rev);
 18
 19  end;
 20  /
```

Enter value for n: 35456

old 8: n:=&n;

new 8: n:=35456;

reverse is 65453

PL/SQL procedure successfully completed.

3. Write a PL/SQL function to reverse a given string.

```
1  CREATE OR REPLACE FUNCTION reverse (
2      string_in IN VARCHAR2
3  )
4      RETURN VARCHAR2
5  IS
6      l_position  integer := 1;
7      l_length    integer := NVL (LENGTH (string_in), 0);
8      l_return    VARCHAR2 (100);
9  BEGIN
10     WHILE (l_position <= l_length)
11     LOOP
12         l_return := SUBSTR (string_in, l_position, 1) ||
                    l_return;
```



```

13         l_position := l_position + 1;
14     END LOOP;
15     RETURN l_return;
16*  END reverse;
SQL> /

```

Function created.

```
SQL> select reverse('abcd') from dual;
```

```
REVE
```

```
----
```

```
dcba
```

4. Write a PL/SQL code to raise the salary of employee by 10% where their designation is 'Assistant Professor'.

```
SQL> set serveroutput on;
```

```
SQL> select * from worker;
```

EMPNO	ENAME	DESIGNATION	SALARY
100	Ravi	Asst_Prof	10000
103	Priya	Professor	15000
105	Harsha	Asst_Prof	12000
106	Divya	HOD	50000

```

1  begin
2      update worker set salary=salary+salary*0.1 where
        Designation = 'Asst_Prof';
3      dbms_output.put_line(sql%rowcount);
4*  end;
5  /

```

```
2
```

PL/SQL procedure successfully completed.

```
SQL> select * from worker;
```

EMPNO	ENAME	DESIGNATION	SALARY
100	Ravi	Asst_Prof	11000
103	Priya	Professor	15000
105	Harsha	Asst_Prof	13200
106	Divya	HOD	50000

5. Create a package to perform insert and delete operation on to the employee table.

```
1  create or replace package pack2 as
2  procedure add_employee(eno number,name varchar2,designation
                        varchar2,salary number);
3  procedure rem_emp(eno number);
4* end;
SQL> /
```

Package created.

```
1  create package body pack2 as
2  procedure add_employee(eno number,name varchar2,designation
varchar2,salary number) is
3  begin
4  insert into worker(Empno,Ename,Designation,Salary)
values(eno,name,designation,salary);
5  end add_employee;
6  procedure rem_emp(eno number) is
7  begin
8  delete from worker where Empno=eno;
9  end rem_emp;
10* end pack2;
SQL> /
```

Package body created.

```
1  declare
2  begin
3  pack2.add_employee('111','James','professor','35000');
4  pack2.add_employee('222','Robert','Assistant
professor','45000');
5  pack2.add_employee('333','Jones','Associate
professor','55000');
6  pack2.rem_emp('111');
7* end;
SQL> /
```

PL/SQL procedure successfully completed.

```
SQL> select * from worker;
```

EMPNO	ENAME	DESIGNATION	SALARY
100	Ravi	Asst_Prof	11000
103	Priya	Professor	15000
105	Harsha	Asst_Prof	13200
106	Divya	HOD	50000
222	Robert	Assistant professor	45000
333	Jones	Associate professor	55000

6 rows selected.

6. Write a PL/SQL procedure to check whether the eno=1001 exist. If it is not raise an exception.

```

1  DECLARE
2      e_no worker.empno%type := 1001;
3      e_name worker.Ename%type;
4      e_sal worker.Salary%type;
5      e_designation worker.Designation%type;
6  BEGIN
7      SELECT Empno,Ename, Designation,Salary INTO
e_no,e_name,e_sal,e_designation
8      FROM worker
9      WHERE Empno = e_no;
10  DBMS_OUTPUT.PUT_LINE ('Name: ' || e_name);
11  DBMS_OUTPUT.PUT_LINE ('Salary: ' || e_sal);
12  EXCEPTION
13      WHEN no_data_found THEN
14          dbms_output.put_line('No such employee number!');
15      WHEN others THEN
16          dbms_output.put_line('Error!');
17* END;
SQL> /
No such employee number!

```

PL/SQL procedure successfully completed.

Ex. : 08 PL / SQL : CURSORS
Date : 21/02/2019

Aim:

- 1. Write a PL/SQL program using cursors to print empname, designation, salary, experience, depending on the criteria given below:**
 - a. If the employee has worked for <1 yr raise an exception not to display his details.**
 - b. If the employee has worked for >1 yr and <2 yr increase salary by .15%**
 - c. If the employee has worked for >2 yrs increase salary by .2%**

SQL> select * from employee;

EMP_NO	EMP_NAME	DESIGNATIO	SALARY	DOJ
100	ishwarya	Professor	1000	29-APR-18
110	Alex	Asst_Prof	4626	28-MAY-17
120	vishal	Professor	15000	11-MAR-16
130	dhanush	Assoc_Prof	20640	20-NOV-10

```

1  declare
2  c_empname employee.emp_name%type;
3  c_empno employee.emp_no%type;
4  c_salary employee.salary%type;
5  c_design employee.designation%type;
6  c_doj employee.doj%type;
7  m number(5);
8  ex exception;
9  cursor emp_cur IS
10 select emp_no,emp_name,designation,salary,doj from
employee;
11 begin
12 OPEN emp_cur;
13 begin
14 LOOP
15 begin
16 fetch emp_cur into
c_empno,c_empname,c_design,c_salary,c_doj;
17 exit when emp_cur%notfound;
18 IF(months_between(sysdate,c_doj))<12 then
19     raise ex;
20 continue;
21 ELSIF(((months_between(sysdate,c_doj))<24) and
((months_between(sysdate,c_doj))>12)) then

```

```
22      c_salary:=c_salary+c_salary*0.15;
23      update employee set salary=c_salary where
(months_between(sysdate,employee.doj))<24 and
(months_between(sysdate,employee.doj))>12;
24      dbms_output.put_line('Employee name: ' || c_empname ||
        'Designation: ' || c_design || 'Salary: ' || c_salary || 'Date
        of join' || c_doj);
25      continue;
26  ELSE
27      c_salary:=c_salary+c_salary*0.2;
28      update employee set salary=c_salary where
(months_between(sysdate,employee.doj))>24 and emp_no=c_empno;
29      dbms_output.put_line('Employee name: ' || c_empname ||
        'Designation: ' || c_design || 'Salary: ' || c_salary || 'Date
        of join' || c_doj);
30      continue;
31  END IF;
32  exception
33      when no_data_found then
34          dbms_output.put_line('ERROR: no data found!');
35      when ex then
36          dbms_output.put_line('Employee: ' || c_empname || ' has
        less than a year experience! Cannot display data!');
37          continue;
38  end;
39  end loop;
40  end;
41  close emp_cur;
42* end;
SQL> /
```

Employee: ishwarya has less than a year experience! Cannot display data!

Employee name: Alex Designation: Asst_Prof Salary: 5320 Date of join: 28-MAY-17

Employee name: vishal Designation: Professor Salary: 18000 Date of join: 11-MAR-16

Employee name: dhanush Designation: Assoc_Prof Salary: 24768 Date of join: 20-NOV-10

PL/SQL procedure successfully completed.

```
SQL> select * from employee;
```

EMP_NO	EMP_NAME	DESIGNATIO	SALARY	DOJ
100	ishwarya	Professor	1000	29-APR-18
110	Alex	Asst_Prof	5320	28-MAY-17
120	vishal	Professor	18000	11-MAR-16
130	dhanush	Assoc_Prof	24768	20-NOV-10

2. The HRD manager has decided to raise the salary for all the employees in department no:20 by .05% whenever such a raise is given to the employees the record for the same are maintained in emprise table. It includes eno, date when the raise was given and actual raise. Write a PL/SQL program to update sal of each and insert record in emprise table.

```
select * from employee;
```

EMP_NO	EMP_NAME	DESIGNATIO	SALARY	DOJ	DEPT_NO
100	ishwarya	Professor	1000	29-APR-18	10
110	Alex	Asst_Prof	5320	28-MAY-17	20
120	vishal	Professor	18000	11-MAR-16	30
130	dhanush	Assoc_Prof	24768	20-NOV-10	40

```
create table emprise(empno number(5),raise_date date,raise_amt
number(5));
```

Table created.

```

1  declare
2  c_empno employee.emp_no%type;
3  c_empname employee.emp_name%type;
4  c_salary employee.salary%type;
5  c_deptno employee.dept_no%type;
6  cursor emp_cur IS
7  select emp_no,emp_name,salary,dept_no from employee;
8  begin
9  open emp_cur;
10 loop
11 fetch emp_cur into c_empno, c_empname, c_salary, c_deptno ;
12 exit when emp_cur%notfound;
13 if(c_deptno=20) THEN
14 insert into emprise (empno,raise_date,raise_amt)
values(c_empno,sysdate,c_salary*0.05);
15 c_salary:=c_salary + c_salary*0.05;
16 update employee set employee.salary=c_salary where
dept_no=20;
17 dbms_output.put_line('employee no: '||c_empno||' employee
name: '||c_empname||' Salary: '||c_salary||' dept no:' ||
```

```

c_deptno);
18  end if;
19  end loop;
20  close emp_cur;
21* end;
SQL> /
employee no: 110 employee name: Alex Salary: 5586 dept no: 20

```

PL/SQL procedure successfully completed.

```
SQL> select * from empraise;
```

EMPNO	RAISE_DAT	RAISE_AMT
110	08-MAR-19	266

```
SQL> select * from employee;
```

EMP_NO	EMP_NAME	DESIGNATIO	SALARY	DOJ	DEPT_NO
100	ishwarya	Professor	1000	29-APR-18	10
110	Alex	Asst_Prof	5586	28-MAY-17	20
120	vishal	Professor	18000	11-MAR-16	30
130	dhanush	Assoc_Prof	24768	20-NOV-10	40

3. Write a program using cursor to display the details of employees whose sum of sal and comm. Rs>6000

```
SQL> select * from employee;
```

EMP_NO	EMP_NAME	DESIGNATIO	SALARY	DOJ	DEPT_NO	COMM
100	ishwarya	Professor	1000	29-APR-18	10	1000
110	Alex	Asst_Prof	5586	28-MAY-17	20	200
120	vishal	Professor	18000	11-MAR-16	30	1500
130	dhanush	Assoc_Prof	24768	20-NOV-10	40	1200

```

1  declare
2  c_empno employee.emp_no%type;
3  c_empname employee.emp_name%type;
4  c_salary employee.salary%type;
5  c_comm employee.comm%type;
6  myitem employee%rowtype;
7  cursor emp_cur IS
8  select emp_no,emp_name,salary,comm FROM employee;
9  begin
10 open emp_cur;

```

```

11  loop
12  fetch emp_cur into c_empno,c_empname,c_salary,c_comm;
13  exit when emp_cur%NOTFOUND;
14  if(c_salary + c_comm > 6000)
15  THEN
16  dbms_output.put_line('Employee no: '||c_empno || ' Employee
name: ' || c_empname || ' Employee salary: ' ||c_salary || '
Comission'||c_comm);
17  end if;
18  end loop;
19  close emp_cur;
20* end;
SQL> /

```

Employee no: 120 Employee name: vishal Employee salary: 18000
Comission:1500

Employee no: 130 Employee name: dhanush Employee salary: 24768
Comission:1200

PL/SQL procedure successfully completed.

4. Write a program to find the name, salary of employee with netsal(includes sal and comm.) if netsal>6000 display name, salary otherwise throw an exception.

```

1  declare
2  c_empno employee.emp_no%type;
3  c_empname employee.emp_name%type;
4  c_salary employee.salary%type;
5  c_comm employee.comm%type;
6  cursor emp_cur IS
7  select emp_no,emp_name,salary,comm from employee;
8  ex exception;
9  begin
10  open emp_cur;
11  loop
12  begin
13  fetch emp_cur into c_empno,c_empname,c_salary,c_comm;
14  exit when emp_cur%notfound;
15  if(c_salary + c_comm > 6000) then
16  dbms_output.put_line('Employee no: '||c_empno ||
'Employee name: ' || c_empname || ' Employee salary: '
||c_salary|| ' Employee comission '||c_comm ||' Net salary:
'|(c_salary+c_comm));
17  else
18  raise ex;
19  end if;
20  exception

```



```

21         when ex then
22             dbms_output.put_line(c_empname|| ' ERROR! ');
23     continue;
24 end;
25 end loop;
26 close emp_cur;
27* end;
SQL> /

```

ishwarya ERROR!

Alex ERROR!

Employee no: 120 Employee name: vishal Employee salary: 18000

Employee commission: 1500 Net salary: 19500

Employee no: 130 Employee name: dhanush Employee salary: 24768

Employee commission: 1200 Net salary: 25968

PL/SQL procedure successfully completed.

5. Write a program to increment the salary of all employees by Rs.2000 who belong to deptno=10 and display the number of rows updated using implicit cursors.

```

1  declare
2  c_empno employee.emp_no%type;
3  c_empname employee.emp_name%type;
4  c_salary employee.salary%type;
5  c_deptno employee.dept_no%type;
6  c_comm employee.comm%type;
7  cursor emp_cur IS
8  select emp_no,emp_name,salary,dept_no,comm from employee;
9  begin
10 open emp_cur;
11 loop
12 fetch emp_cur into c_empno, c_empname, c_salary, c_deptno,
c_comm;
13 exit when emp_cur%notfound;
14 if(c_deptno=10) then
15     c_salary:=c_salary + 2000;
16     update employee set employee.salary=c_salary where
dept_no='10';
17     dbms_output.put_line('employee no: '||c_empno||'
employee name: '||c_empname||' Salary: '||c_salary||' dept no:
'||c_deptno);
18 dbms_output.put_line('The number of rows updated is:
'||sql%rowcount);
19 end if;
20 end loop;
21 close emp_cur;

```

```
22* end;
SQL> /
employee no: 100 employee name: ishwarya Salary: 3000 dept no:10
The number of rows updated is: 1
```

```
SQL> select * from employee;
```

EMP_NO	EMP_NAME	DESIGNATIO	SALARY	DOJ	DEPT_NO	COMM
100	ishwarya	Professor	3000	29-APR-18	10	1000
110	Alex	Asst_Prof	5586	28-MAY-17	20	200
120	vishal	Professor	18000	11-MAR-16	30	1500
130	dhanush	Assoc_Prof	24768	20-NOV-10	40	1200

Ex. : 09 PL / SQL : TRIGGERS

Date : 05/03/2019

Aim:

1. Create a trigger for emp table which makes the entry in name column in upper case.

```

1  create or replace trigger trig1
2      after insert on emp
3      begin
4          update emp set emp_name = upper(emp_name);
5          dbms_output.put_line('Upper Cased');
6* end;
SQL> /

```

Trigger created.

```

SQL> insert into emp values ('150' , 'Sunil', 'Accountant',
'15000');
Upper Cased

```

1 row created.

```

SQL> select * from emp;

```

EMP_NO	EMP_NAME	DESIGNATIO	SALARY
-----	-----	-----	-----
150	SUNIL	Accountant	15000

2. Create a trigger for inserting, updating records into emp table and sal should be greater than 10000.

```

1  create or replace trigger trig2
2      before insert or update on emp
3      for each row
4      when(new.salary>10000)
5      begin
6          dbms_output.put_line('Salary is greater than 10000');
7* end;
SQL> /

```

Trigger created.

```
SQL> insert into emp values('222','Sakthi','Professor','14000');
```

Salary is greater than 10000

1 row created.

```
SQL> select * from employee;
```

EMP_NO	EMP_NAME	DESIGNATIO	SALARY
150	SUNIL	Accountant	15000
222	SAKTHI	Professor	14000

2 rows selected.

3) Create a trigger on emp table that avoids deletion on Fridays.

```
1 create or replace trigger trig3
2   before delete on employee
3   for each row
4   when(to_char(sysdate,'D')=6)
5   begin
6     raise_application_error(-20000,'Deletion not
accessed on this day');
7* end;
SQL> /
```

Trigger created.

```
SQL> delete from employee where emp_no='222';
delete from employee where emp_no='222'
      *
```

ERROR at line 1:

ORA-20000: Deletion not accessed on this day

ORA-06512: at "HR.TRIG3", line 2

ORA-04088: error during execution of trigger 'HR.TRIG3'

4. Create a trigger on emp table so that records of 'sunil' not be deleted.

```
1 create or replace trigger trig4
2   before delete on employee
3   for each row
4   when(old.emp_name='SUNIL')
5   begin
6     raise_application_error(-20000,'The name is Sunil.
Deletion not possible');
```

```
7* end;  
SQL> /
```

Trigger created.

```
SQL> delete from employee where emp_no='111';  
delete from employee where emp_no='111'  
*
```

ERROR at line 1:

ORA-20000: The name is Sunil. Deletion not possible

ORA-06512: at "HR.TRIG4", line 2

ORA-04088: error during execution of trigger 'HR.TRIG4'