Name: Amruta Vilas Shinde

Roll-no: 219445

Class: SY.Bsc.IT (Semester 3)

Subject: Database Management Systems Practicals

INDEX

Amruta Shinde
219445

GoodLuck | Page No.
Date

Sr	Date	Topic:-
1	8-9-21 9-9-21	SQL Statements-I

A] SQL Statements.

Consider a relation EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO)

Basic SQL-Query

1) Find the details of all the employees

SQL> select * from emp;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30

13 rows selected
SQL>

2. Display the name and salary of the employees

SQL> Select ename, sal from emp;

ENAME	SAL
SMITH	800
ALLEN	1600

13 rows selected
SQL>

Amruta Shinde
219445

GoodLuck | Page No.
Date

3. Display the name of all the employees

SQL> select ename from emp;

ENAME
SMITH
ALLEN

13 rows selected
SQL>

4) Find the salary of all employees incremented by 2000

SQL> select sal+2000 as "NEW SAL" from emp;

NEW SAL
2000
3600

13 rows selected
SQL

5 Display the details of employees when salary is greater than 2000

SQL> select * from emp where sal > 2000;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7566	JONES	MANAGER	7839	2-APR-81	2975		20
7698	BLAKE	MANAGER	7839	1-NOV-81	2850		20

Amruta Shinde
219445

GoodLuck | Page No.
Date

6 Display the name of the employee when salary range bet^n 2000 & 5000

SQL> select ename from emp where sal bet^n 2000 & 5000

ENAME
JONES
BLAKE

7. Display the details when the Job of employee is either Marketing salesman, and Analyst

SQL> Select EMPNO, ENAME, JOB from EMP where JOB IN ('MARKETING', 'SALESMAN', ANANYST)

EMPNO	INAME	JOB
7499	ALLEN	SALESMAN
7521	WARD	SALESMAN

8. Find the name of employees when the name starts with A

SQL> select ename from emp where ename like 'A%';

ENAME
ALLEN
ADAMS

9 Display the details when commission is NULL

SQL> select * ~~emp~~ from emp where comm is null;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7566	JONES	MANAGER	7839	2-Apr-81	2975		20

Amruta Shinde
219445

GoodLuck | Page No.
Date

10 Display all the details of an employee when job is CLERK and salary is 950

```
SQL> select * from emp where JOB='CLERK' and sal=950;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7900	JAMES	CLERK	7698	3-DEC-01	950		30

11 Display the employee id when name of employee starts with K & is working as manager with salary less than 57000

```
SQL> Select empno, JOB, SAL from emp
     where ename like 'K%' And Job='MANAGER' OR Sal<57000;
```

EMPNO	JOB	SAL
7369	CLERK	800
7499	MANAGR	2975
7521	CLERK	3500
7566	CLERK	4500

③ RESTRICTING AND SORTING DATA:-

1) Display the name of employee is ascending order

```
SQL> Select ename from emp EMP by order by ename;
ENAME
ADAMS
ALLEN
BLAKE
CLARK
```

13 rows selected

2) List of details of employee according to their increasing order of salary

SQL> select * from emp order by sal desc;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	7566	17-NOV-01	5000		10
7700	SCOTT	ANALYST	7539	02-APR-01	3000		20

C] SINGLE ROW FUNCTION

i) NUMERIC FUNCTION (abs, ceil, floor, trunk, round)

consider dual table: DUAL

- DUAL is a dummy table that is automatically created by oracle database
- DUAL table is present under SYS user but accessible to all the user
- DUAL table has one DUMMY column defined with varchar2(1) & contains one row with value X
- selecting from dual table is useful to calculate constant expression with select statement
- you can select a constant, pseudo column or expression from any table, but the value will be reffered as many times as those are rows in the table

1) abs():- Absolute value function:-

ABS() function is used to get the absolute value of number passed as an argument

Syntax:- ABS(expression)

Example:-
To get the absolute value of the number -17.36 & from the DUAL table, the following SQL statement can be used

1) SELECT ABS(-17.36) FROM dual;

ABS(-17.36)
17.36

2) SQL> select abs(-12) from dual;

ABS(-12)
12

3. SQL> select abs(12) from dual;

ABS(12)
12

b) Ceil():- The CEIL() function returns the smallest integer value that is bigger than or equal to a number
Syntax:- CEIL(number)
Parameter Values

Parameter	Description
number	Required A numeric value

Example:-

SQL> select ceil(12.26) from dual;

CEIL(12.26)
13

SQL>select ceil(-10.26) from dual;

CEIL(-10.26)
-10

SQL>select ceil(-10.66) from dual;

CEIL(-10.66)
-10

SQL> select ceil(10.66) from dual;

CEIL(10.66)
11

c] Floor(): The FLOOR() function returns the largest integer value that is smaller than or equal to a number

Syntax: - FLOOR(number)

Parameter Values

Parameter	Description
number	Required. A numeric value

Example:

SQL> select floor(-10.26) from dual;

FLOOR(-10.26)
-11

SQL> select floor(10.26) from dual;

FLOOR(10.26)
10

SQL> select floor(10.76) from dual;

FLOOR(10.76)
10

SQL> select floor(-10.76) from dual;
FLOOR(-10.76)
-11

d] trunc():- The TRUNCATE() function truncate a number to the specific number of decimal places
Syntax:- TRUNCATE(number,decimal)
Parameter Values

Parameter	Description
number	Required. The number to be truncated
decimal	Required. The number of decimal places to truncate to

Example:-
SQL> select trunc(-55.1) from dual;

TRUNC(-55.1)
-55

SQL> select trunc(2.465,1) from dual;

TRUNC(2.465,1)
2.4

SQL> select trunc(142.465,-2) from dual;
TRUNC(142.465,-2)
100

SQL> select trunc(142.465,2) from dual;

TRUNC(142.465,2)
142.46

SQL> select trunc(55,2) from dual;

TRUNC(55,2)
55

SQL> select trunc(55,1) from dual;

TRUNC(55,1)
55

e] Round():- The ROUND() function rounds a number to a specified number of decimal places

Syntax:- ROUND(number, decimals)

Parameter Values

Parameter	Description
number	Required. The number to be rounded
decimals	Optional. The number of decimal places to round number to. If omitted, it returns the integer (no decimals)

Example:-

SQL> select round(34.4158,-1) from dual;

ROUND(34.4158,-1)
30

Note:- The ~~abve~~ above statement will round 34.4158 from the left of the decimal place up to one place

SQL> select round (34.4158, 1) from dual;

ROUND(34.4158,1)
34.4

SQL> select round (55,1) from dual;

ROUND(55,1)
55

SQL> Select round (55, -1) from dual;

ROUND(55,-1)
60

SQL> select round (55.78, -1) from dual;

ROUND(55.78,-1)
60

ii) Character functions [lower, upper, Initcap, ltrim, rtrim, trim, substr, length, lpad]

1) lower() :- The lower returns an specified character ~~in t~~ expression in letter.

Syntax:- lower (c expression)

where c expression → is a given character expression

Query:- SQL> select lower (ENAME) from emp;

Output:-

ENAME
smith
allen
jones
blake
adams

Amruta Shinde
219445

GoodLuck | Page No.
Date

2) Upper(): The upper returns the specified character expression in upper case
Syntax: upper(string1)
where string1 is the given character expression
Query: SQL>select upper(ENAME) from emp;
output:-

ENAME
SMITH
ALLEN
JONES
BLAKE
KING

3) Initcap(): The initcap() sets the 1st letter each each word in upper case & all the letters in lower case
Syntax: initcap(string1)
where string1 whose 1st character will be converted to uppercase and rest character in lower case
Query:-SQL> select initcap('steven king') "Emp name"
from dual;

output
Emp name
Steven name

4) Ltrim(): Ltrim() is used to remove all specified character from the left end side of a string
Syntax: ltrim(string1 [set])
where string1 is the string to trim the characters from left hand side set (optional) It is used for triming
Query : SQL> select ltrim('w3r') "Str" from dual;

output

str
w3r

Query:- SQL> Select ltrim ('000985', '0') "str" from dual;

str
985

5) rtrim(): It is used to remove spaces or set of character which are matching with the triming text from the end of a string
If you do not specify triming text then it defaults to a single blank
syntax: rtrim (string [triming_text])
where string is used to trim the character from right side and triming_text is used for triming

SQL> select rtrim ('w3r') "str" from dual;

str
w3r

left space is there

SQL> select rtrim ('00098 5000', '0') "str" from dual;

str
00098 5

SQL> select rtrim ('0101009850', '01') "str" from dual;

str
0101009

6) trim() : It is used to remove all leading or trailing character or both from a character string

~~It is used to remove~~

: trim([{{leading | trailing | both}}
[trim-character] ~~string~~
/ trim-characters
} from
]
trim-source

SQL> select trim[leading '0' from '000123') from dual;

TRI
123

SQL> Select trim (trailing '0' from '123000') from dual;

TRI
123

SQL> select trim('removing leading and trailing white space') from dual;

TRIM('REMOVINGLEADINGANDTRAILINGWHITESPACE' removing leading and trailing white space

7) Substr() : The substr() function returns the specified number (substr_length) of character from a particular position of given string.
If the position is 0, then it will be treated as 1.
If the position is positive then it counts from begining of character to

Amruta Shinde
219445

GoodLuck | Page No.
Date

(14)

If the position is -ve then it count backward from end of char.
If substr-length is omitted then it returns all character to end of char & if it is less than 1 than it will return null value

:- { substr
| substr B
| substrC
| substr 2
| substr 4
}
(char, position [, substr_length])

SQL> select substr ('w3 resource', 3,4) "substring" from dual;

position — no. of character

subs
reso

SQL> select substr('w3 resource', 5, 6) "substring" from dual;

substr
source

SQL> select substr ('w3resource', -5, 6) "substring" from dual;

subst
ource

8) length(): It is used to return the length of a given string. If string has data type char then length() include all trailing blanks
If a string is null then the function return null
Syntax : { length
length B
length C
length 2
length 4
}
(string 1)

SQL> select length ('w3resource.com') "length in characters"
from dual;

length in characters
14

SQL> select length ('w3resource.com ') "length in character"
from dual;

length in character
16

SQL> select length (' w3resource.com ') "length in character"
from dual

length in character
18

Amruta Shinde
219445

GoodLuck | Page No.
Date

16

9] lpad [left padding]

The lpad() is used to padding the left side of a string with a specified set of character. The function is useful for formating the output of query

Syntax : lpad (expr1, n [, expr2])

SQL> select lpad ('oracle', 4, '*') from dual;

LPAD
orac

Sr no	Date	Topic:
2	4-8-21 5-8-21	Joins SQl Statement - II

9] Displaying Data from Multiple Tables

1) INNER JOIN:-

```
Select empno, ename, sal, dname
from emp inner join dept
on emp.deptno = dept.deptno;
```

output:-

EMPNO	ENAME	SAL	DNAME
7839	KING	5000	ACCOUNTING
7698	BLAKE	3078	SALES
7782	CLARK	2646	ACCOUNTING
7566	JONES	3213	RESEARCH
7788	SCOTT	3000	RESEARCH
7902	FORD	3000	RESEARCH
7369	SMITH	800	RESEARCH
7499	ALLEN	1600	SALES
7521	WARD	1250	SALES
7654	MARTIN	1250	SALES
7844	TURNER	1500	SALES
7876	ADAMS	1100	RESEARCH
7900	JAMES	1350	SALES
7934	MILLER	1300	ACCOUNTING

14 rows selected

2) EqEQUI JOIN :-

```
select empno, ename, sal, dname
from emp, dept
where emp, dept
where emp.deptno = dept.deptno;
```

EMPNO	ENAME	SAL	DNAME
7839	KING	5000	ACCOUNTING
7698	BLAKE	3078	SALES
7782	CLARK	2646	ACCOUNTING
7566	JONES	3213	RESEARCH
7788	SCOTT	3000	RESEARCH
7902	FORD	3000	RESEARCH
7369	SMITH	806	RESEARCH
7499	ALLEN	1600	SALES
7521	WARD	1250	SALES
7654	MARTIN	1250	SALES
7844	TURNER	1500	SALES
7876	ADAMS	1100	RESEARCH
7900	JAMES	1350	SALES
7934	MILLER	1300	ACCOUNTING

14 rows selected

3) OUTER JOIN :-

1. Left Outer Join

```
select empno, ename, sal, dname, loc
from emp left outer join dept
on emp.deptno = deptno; dept.deptno;
```

EMPNO	ENAME	SAL	DNAME	LOC
7934	MILLER	1300	ACCOUNTING	NEW YORK
7782	CLARK	2646	ACCOUNTING	NEW YORK
7839	KING	5000	ACCOUNTING	NEW YORK
7876	ADAMS	1100	RESEARCH	DALLAS
7369	SMITH	800	RESEARCH	DALLAS
7902	FORD	3000	RESEARCH	DALLAS
7788	SCOTT	3000	RESEARCH	DALLAS
7566	JONES	3213	RESEARCH	DALLAS
7900	JAMES	1350	SALES	CHICAGO
7844	TURNER	1500	SALES	CHICAGO
7654	MARTIN	1250	SALES	CHICAGO
7521	WARD	1250	SALES	CHICAGO
7499	ALLEN	1600	SALES	CHICAGO
7698	BLAKE	3078	SALES	CHICAGO

14 rows selected

2. Right Outer Join:-

```
select empno, ename, sal, dname, loc
from emp right outer join dept
on emp.deptno = dept.deptno;
```

EMPNO	ENAME	SAL	DNAME ~~LOC~~	LOC
7839	KING	5000	ACCOUNTING	NEW YORK
7698	BLAKE	3078	SALES	CHICAGO
7782	CLARK	2646	ACCOUNTING	NEW YORK
7566	JONES	3213	RESEARCH	DALLAS
7788	SCOTT	3000	RESEARCH	DALLAS
7902	FORD	3000	RESEARCH	DALLAS
7369	SMITH	800	RESEARCH	DALLAS
7499	ALLEN	1600	SALES	CHICAGO

7521	WARD	1250	SALES	CHICAGO
7654	MARTIN	1250	SALES	CHICAGO
7844	TURNER	1500	SALES	CHICAGO
7876	ADAMS	1100	RESEARCH	DALLAS
7900	JAMES	1350	SALES	CHICAGO
7934	MILLER	1300	ACCOUNTING	NEW YORK
			OPERATIONS	BOSTON

15 rows selected

3. Full Outer Join

```
select empno, ename, sal, dname, loc
from emp full outer join dept
on emp.deptno = dept.deptno;
```

EMPNO	ENAME	SAL	DNAME	LOC
7839	KING	5000	ACCOUNTING	NEW YORK
7698	BLAKE	3078	SALES	CHICAGO
7782	CLARK	2646	ACCOUNTING	NEW YORK
7566	JONES	3213	RESEARCH	DALLAS
7788	SCOTT	3000	RESEARCH	DALLAS
7902	FORD	3000	RESEARCH	DALLAS
7369	SMITH	400	RESEARCH	DALLAS
7499	ALLEN	1600	SALES	CHICAGO
7521	WARD	1250	SALES	CHICAGO
7654	MARTIN	1250	SALES	CHICAGO
7844	TURNER	1500	SALES	CHICAGO
7876	ADAMS	1100	RESEARCH	DALLAS
7900	JAMES	1350	SALES	CHICAGO
7934	MILLER	1300	ACCOUNTING	NEW YORK
			OPERATIONS	BOSTON

15 rows selected

4. Cross Join

```
select empno, dname
from emp
cross join dept;
```

EMPNO	DNAME
7369	ACCOUNTING
7499	ACCOUNTING
7521	ACCOUNTING
7566	ACCOUNTING
7654	ACCOUNTING
7698	ACCOUNTING
7782	ACCOUNTING
7788	ACCOUNTING
7839	ACCOUNTING
7844	ACCOUNTING
7876	ACCOUNTING
7900	ACCOUNTING
7902	ACCOUNTING
7934	ACCOUNTING
7369	ACCOUNTING
7499	RESEARCH
7521	RESEARCH
7566	RESEARCH
7654	RESEARCH
7698	RESEARCH
7782	RESEARCH
7788	RESEARCH
7839	RESEARCH
7844	RESEARCH
7876	RESEARCH
7900	RESEARCH
7902	RESEARCH
7934	RESEARCH
7369	SALES
7499	SALES
7521	SALES
7566	SALES
7654	SALES
7698	SALES
7782	SALES
7788	SALES
7839	SALES
7844	SALES
7876	SALES
7900	SALES
7902	SALES
7934	SALES
7369	OPERATION
7499	OPERATION
7521	OPERATION
7566	OPERATION
7654	OPERATION
7698	OPERATION
7782	OPERATION
7788	OPERATION
7839	OPERATION
7844	OPERATION

7876 OPERATIONS
7900 OPERATIONS
7902 OPERATIONS
7934 OPERATIONS
56 rows selected

5) Natural Join

```
select*
from emp natural join dept;
```

DEPTNO EMPNO ENAME JOB MGR HIREDATE SAL

DEPTNO	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DNAME	LOC
10	7839	KING	PRESIDENT		17-NOV-81	500		ACCOUNTING	NEW YORK
30	7698	BLAKE	MANAGER	7839	1-MAY-81	3078		SALES	CHICAGO
10	7782	CLARK	MANAGER	7839	09-JUN-81	2646		ACCOUNTING	NEW YORK
20	7566	JONES	MANAGER	7839	02-APR-81	3213		RESEARCH	DALLAS
20	7788	SCOTT	ANALYST	7566	19-APR-87	3000		RESEARCH	DALLAS
20	7902	FORD	ANALYST	7566	03-DEC-81	3000		RESEARCH	DALLAS CHICAGO
20	7369	SMITH	CLERK	7902	17-DEC-80	800		RESEARCH	DALLS
30	7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	SALES	CHICAGO
30	7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	SALES	CHICAGO
30	7654	MARTIN	SALESMAN	7698	28-SEP-81	1200	1400	SALES	CHICAGO
30	7844	TURNER	SALESMAN	7698	8-SEP-81	1500	0	SALES	CHICAGO
20	7876	ADAMS	CLERK	7788	23-MAY-87	1100		RESEARCH	DALLAS
30	7900	JAMES	CLERK	7698	03-DEC-81	1350		SALES	CHICAGO
10	7934	MILLER	CLERK	7782	23-JAN-82	1300		ACCOUNTING	NEW YORK

B] Aggregating Data Using Group Functions:-
Aggregating function is always use with group by function
Aggregating functions are of various type
e.g ~~avg()~~ avg(), min(), max(), total(), sum(), count()

consider emp table and perform use of Aggrigate function

1 select deptno, min(sal), max(sal), count(*)
from emp
group by deptno;

Output:

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	1250	3078	6
20	800	3213	5
10	1300	5000	3

2 select deptno, min(sal), max(sal), count(*)
from emp
where sal>1000
group by deptno;

output:-

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	1250	3078	6
20	1100	3213	4
10	1300	5000	3

3 select deptno, min(sal), max(sal), count(*)
from emp
where ename like 'J%'
group by deptno;

output:-

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	1350	1350	1
20	3213	3213	1

4. select deptno, min(sal), max(sal), count(*)
from emp
where ename like 'J%'
group by deptno
order by deptno;

Output

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
20	3213	3213	1
50	1350	1350	1

5 select deptno, min(sal), max(sal), count(*)
from emp
where sal > 1000
group by deptno
having min(sal) > 1000;

Output:-

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	1250	3078	6
20	1100	3213	4
10	1300	5000	3

Amruta Shinde
219445

C] Subquery

1] SQL> select ename
from emp
where sal > (select sal
from emp
where empno = 7566);

output:

ENAME
SCOTT
King
Turner
ford

2] SQL> select ename, sal, deptno, job
from emp
where job =
(select job
from emp
where empno = 7369);

ENAME	SAL	DEPTNO	JOB
SMITH	800	20	CLERK
ADAMS	1100	20	CLERK
JAMES	950	30	CLERK
MILLER	1300	10	CLERK

3) SQL> select ename, sal, deptno
from emp
where sal in
(select min(sal)
from emp
group by deptno);

Amruta Shinde
21944S

output

ENAME	SAL	DEPTNO
SMITH	800	20
JAMES	950	30
MILLER	1300	10

a
```
SQL> select empno, ename, job
     from emp
     where sal<ANY
          (select sal
          from emp
          where job='CLERK');
```

EMPNO	ENAME	JOB
7369	SMITH	CLERK
7900	ADAMS	CLERK
7876	JAMES	CLERK
7521	WARD	SALESMAN
7654	MARIN	SALESMAN

b
```
SELECT empno, ename, job
FROM EMP
WHERE sal > ALL
          (SELECT sal
          FROM emp
          where job='CLERK')
          AND job <> 'CLERK';
```

EMPNO	ENAME	JOB
7844	TURNER	SALESMAN
7499	ALLEN	SALESMAN
7782	CLARK	MANGER
7902	FORD	ANALYST

Amruta Shinde
219445

c SELECT empno, ename, job
FROM emp
WHERE sal > ALL
(SELECT AVG(sal)
FROM emp
GROUP BY deptno);

EMPNO	ENAME	JOB
7968	JONES	MANAGER
7788	SCOTT	ANALYST
7639	KING	PRESIDENT

Amruta Shinde
219495

GoodLuck | Page No.
Date

Sr	Date	Topic
3	27-7-21	Manipulating Data

a] Using INSERT statement

```
Create table employee
(enum int primary key,
elname varchar (15),
efname varchar (15),
sal int,
ehdate date,
jobcode char(8));
```

Output:
Table created

Display the structure of employee table

SQL> desc employee;

Name	Null?	Type
ENUM	Not NULL	NUMBER(38)
ELNAME		VARCHAR2(15)
EFNAME		VARCHAR2(15)
SAL		NUMBER(38)
EHDATE		DATE
JOBCODE		CHAR(8)

SQL>

Insert values in employee table

There are 3 ways to insert record in employee table & they are

1) First way:

```
insert into employee values (1,'P','Raman',6000,sydate,
'909');
1 row created
```

2) second way

```
insert into employee (enum,elname,sal) values (2,'R',6000);
1 row created
```

3) Third way

```
insert into employee (&enum,'&elname','&efname',&sal,
&ehdate,'&joc '&jcode');

Enter value of enum: 3
Enter value of elname: L
Enter value for efname: james
Enter value for sal: 9000
Enter value for ehdate: sysdate
Enter value for jcode: 908
old 1: insert into employee values
(&num,'&elname','&efname',&sal,&ehdate,'&jcode')
new 1: insert into employee values (3,'L','james',9000,sysdate,'908')
1 row created

SQL>/
Enter value of enum: 4
Enter value of elname: U
Enter value of efname: david
Enter value for sal: 8000
```

Enter value for ehdate: '12-dec-17'
Enter value for jcode: 888
old 1: insert into employee values
(&enum, '&elname', '&efname', &sal, &ehdate, '&jcode')
new 1: insert into employee values (4, 'U', 'david', 8000, '12-dec-17', '888')

3 Display all the records of employee table
SQL> select* from employee;

ENUM	ELNAME	EFNAME	SAL	EHDATE	JOBCODE
1	P	Raman	6000	26-JUL-21	909
2	R		6000		
3	L	james	9000	30-AUG-17	908
4	U	david	8000	12-DEC-17	888

4 USING DELETE STATEMENTS:
i) Delete all the records from the table employee
delete from employee:
1 row deleted
NOTE:- Only row values are deleted but still table structure, remains
ii) Delete all the records when salary is greater than 1000
delete from employee where sal>1000;
iii) Delete all records when commission is 0
delete from employee where comm=0;

Nam
Amruta shinde
219445

GoodLuck | Page No.
Date

31

5 Using update statement

i) Update the salary of all employees

```
Update employee
set sal = 2000;
4 rows updated
SQL> select * from employee;
```

ENUM	ELNAME	EFNAME	SAL	EHDATE	JOBCODE
1	P	Raman	2000	26-JUL-21	909
2	R		2000		
3	N	Sam	2000	26-JUL-21	809
4	F	gita	2000	12-AUG-20	589

ii) Update the salary of employee when name starts with R

```
Update employee
set sal = 8000
where ~~of~~ EFNAME
where efname like 'R%'
1 row updated
SQL> select * from employee;
ENUM ELNAME EFNAME SAL EHDATE JOBCODE
  1    P     Raman 8000 26-JUL-21 909
```

iii) Update the salary of employee when commission is NULL

```
Update employee
Set sal = 1000
where comm is NULL
1 row updated
SQL> select * from employee;
ENUM ELNAME EFNAME SAL EHDATE JOBCODE
  2    R          1000
```

Amruta Shinde
219445

GoodLuck | Page No.
Date

(32)

Sr No	Date	Topic: Practical 4
4	6-8-21	Constraints

Creating & Managing Tables including Constraints
Create the given table: employee

Attributename	Data declaration	Constraints
emp_num	Int	Primary key
emp_lname	Varchar(15)	
emp_fname	Varchar(15)	not null
emp_sal	Int	
emp_hiredate	date	
Job_code	char(3)	default 'SOL'

1) Create
SQL> Create table employee
(emp_num int primary key,
emp_lname ~~int~~ varchar(15),
emp_fname varchar(15) not null
emp_sal int,
emp_hiredate
Job_code char(3) default 'SOL');
Table created

2 Describe employee table
SQL> desc employee;

Name	NULL	Type
emp_num	not null	number(38)
emp_lname		varchar(15)
emp-fname	not null	varchar(15)
emp-sal		number(38)
emp hiredate		date
Job-code		char(3)

3] alter table employee: add column gender

```
SQL> alter table employee
     add gender char(5);
Table altered
SQL> desc employee;
```

Name	NULL	Type
emp_num	not null	number(38)
emp_lname		varchar(15)
emp-fname	not null	varchar(15)
emp_sal		number(38)
emp hiredate		date
Job_code		char(3)
gender		char(5)

4 Modify column gender with data type (3)

```
SQL> alter table employee
     modify gender char(3)
Table created
SQL> desc employee
```

gender char(3)

5] drop column gender
SQL> alter table employee
drop column gender;
Table altered

SQL> desc employee;
There will be no column gender

6] Insert two valid values in employee table
SQL> insert into employee value(1,'patil','smith', 3000, sysdate, 501);

1 row created

SQL> insert into employee value(2,'dawis','hack',3000, '6-JUN-21','A');

1 row created

7] Find out the details of employee table
SQL> select * from employee;

EMPNUM	EMP_INAME	EMP_ENAME	EMP_SAL	EMP_HIREDATE	JOB
1	Patil	Smith	3000	05-AUG-21	501
2	dawis	jack	800	06-JUL-21	A

8) drop table employee
SQL> drop table employee;
Table dropped

SQL> select * from employee;
select * from employee

ERROR table or view does not exist
SQL> desc employee;
ERROR object employee does not exist

Amruta Shinde
219445

GoodLuck | Page No.
Date / /

(86)

Sr	Date	Topic-Practical 5
25	3-8-21	Creating & Managing other DB object

a] Creating Views & Managing other DB objects:-

a] Creating Views

Note:- write down the steps before doing view practs

SQL>conn
Enter user-name: sys as sysdba
Enter password: press enter
connected
SQL> grant all privileges to scott
2;
Grant succeded
SQL>conn
Enter user-name: scott
Enter password:
connected

1) HORIZONTAL VIEWS:-

SQL>create view vw_hemp
as
select * from emp
where deptno = 10;

View created

SQL> select * from vw_hemp;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7839	KING	PRESIDENT		17-NOV-21	5000		10

2) VERTICAL VIEWS:-

```
SQL> Create view vw_vemp
  as
  select empno, ename, JOB from emp;
view created
SQL> select * from vw_vemp;
```

EMPNO	ENAME	JOB
7369	SMITH	CLERK
7499	ALLEN	SALESMAN
7566	WARD	MANAGER

13 rows selected

3) ROW/COLUMN SUBSET VIEWS:-

```
SQL> create view rc_emp
  as
  select empno, ename, JOB, mgr from emp
  where comm is null;

view created
SQL> select * from rc_emp;
```

EMPNO	ENAME	JOB	MGR
7369	SMITH	CLERK	7902
7566	JONES	MANAGER	7839
7698	BLAKE	MANAGER	7839

4) GROUPED VIEWS:-

```
SQL> create view gp_name(job,total_sal)
 as
 select job,sum(sal)
 from emp
 group by job;
```

View created

SQL> select * from gp_name;

JOB	TOTAL_SAL
CLERK	4150
SALESMAN	5600
PRESIDENT	5000
MANAGER	8275
ANALYST	6000

5) JOINED VIEWS:-

FIRST CHECK DEPT TABLE

SQL> select * from dept;

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

```
SQL> create View
jv_empd(empno, ename, job, deptno, loc)
as
select empno, ename, job, dept.deptno, loc
from emp join dept
on emp.deptno = dept.deptno;
```

View created.

SQL> select * from jv_empd;

EMPNO	ENAME	JOB	DEPTNO	LOC
7782	CLARK	MANAGER	10	NEW YORK
7839	KING	PRESIDENT	10	NEW YORK
7934	MILLER	CLERK	10	NEW YORK
7566	JONES	MANAGER	20	NEW YORK

14 rows selected

REPLACING A VIEW:-

```
SQL> create or replace view vw_emp
as
select empno, ename, sal
from emp
where sal between 1000 and 5000
with check option

View created.
SQL> create or replace view vw_name
as
select * from emp
where deptno=10 and ename like 'M%';

view created
SQL> select * from vw_name;
```

output

EMPNO	ENAME	SAL	JOB	COMM	MGR	HIREDATE	DEPTNO
7934	MILLER	1300	CLERK		778223-	JAN-82	10

Amruta Shinde
219495

GoodLuck | Page No.
Date / /

40

VIEW UPDATES:-

```
SQL> create or replace view vw-emp
as
select empno, ename, sal from emp
where sal between 1000 and 5000
with check option;
```

output:-

view created.

```
SQL> select * from vw_emp;
```

output:-

EMPNO	ENAME	SAL
7499	ALEN	1600
7521	WARD	1250
7566	JONES	2975
7654	MARTIN	1250
7934	MILLER	1300

12 rows selected

Sr.no	Date	Topic:-
6	9/8/21 10/8/21	Practical 6

Using SET operators, Date / Time functions, GROUP BY clause (advanced features) and advanced sub queries

a] Using set operators

Consider two tables with table name Product_details with attributes:-
(pdt_id, pdt_name, qty, price) & another table name sales_details with attributes:-
(sales_no, pdt_id, qty, price, cust name) and then after performs the following operation on it:-

output:- Table created

```
Create table product_details
(pdt_id number(4),
pdt_name varchar2(10),
qty number(5),
price number(6));
```

output Table created

```
create table sales_details
(sales_no number(4),
pdt_id number(4),
qty number(5),
price number(6),
cust name varchar2(10));
```

1 Insert table into both the table

insert into product_details
values (1, 'pendrive', 100, 900);
1 row created

insert into product_details values (2, 'harddisk', 200, 4000);
1 row created

insert into product_details values (3, 'DVD', 20, 1000);
1 row created

insert into product_details value (4, 'speaker', 60, 2400);
1 row created

2 Now inserting values into sales_details in table

insert into sales_details values (101, 1, 50, 900, 'savni');
1 row created

insert into sales_details values (102, 4, 10, 800, 'sam');
1 row created

insert into sales_details values (103, 2, 30, 1000, 'harsh');
1 row cre

3 select find out all the values from sales_
Product_details and sales_details
SQL> select * from product_details:-

PDT_ID	PDT_NAME	QTY	PRICE
2	harddisk	200	14000
3	DVD	20	1000
4	speaker	60	2400
1	Penchive	100	900

SQL> select * from sales_details;

SALES_NO	PDT_ID	QTY	PRICE	CUST_NAME
101	1	50	200	savni
102	4	10	800	sami
103	2	30	1000	harsh

I) Given the information what would be the query output for the following
a) A UNION query based on these two tables
b) A UNION ALL query based on these two tables
c) An INTERSECT query based on these two tables
d) A MINUS query based on these two tables

a) UNION

```
SQL> select pdt_id from product_details
     Union
     select pdb_id from sales_details;
```

output

PDT_ID
1
2
3
4

b) Union all

```
SQL> select pdt_id from product_details
     union all
     select pdt_id from sales_details;
```

output

PDT_ID
2
3
4
1
1
4
2

7 rows selected

c) Intersect

```
SQL> select pdt_id from product_details
     intersect
     select pdt_id from sales_details;
```

output

PDT_ID
1
2
4

d) Minus

```
SQL> select pdt_id from product_details
     minuns
     select pdt_id from sales_details;
```

output

PDT_ID
3

Amruta Shinde
219445

GoodLuck | Page No. 45
Date

b Date|Time Function

~~1) Current_do: It returns the current date~~

1) current_date(): It returns the current date
Syntax: current_date
SQL> SELECT CURRENT_DATE FROM DUAL;

output
CURRENT_D
06-AUG-21

2) Sysdate(): Returns current date for the os on which our database is present
Syntax: sysdate
SQL> SELECT SYSDATE FROM DUAL:

output
SYSDATE
06-AUG-21

3) Add_months(): Returns a date with a given number of months added (date + integer months)
A months is defined by session parameter
Syntax: add_months (date, integer)
where, date is a date time value converted to date, and integer is an integer value

3 month added i.e 6th Aug 2021
system date 06 Aug

SQL> SELECT ADD_MONTHS (SYSDATE, 3) FROM DUAL;

output
ADD_MONTH
06-NOV-21

4) Next_day(): ~~A~~ Returns the date of the first week day therefore later then the date
syntax: next_day(date, char)
SQL> SELECT NEXT_DAY (SYSDATE, 'WEDNESDAY') FROM DUAL;

output
NEXT_DAY
18-MAR-15

5 last_day(): Return the last date of the month that contains a date. the return type is always date, regardless of the datatype
Syntax: last_day(date)
SQL> SELECT last_DAY (sysdate) FROM DUAL;
output LAST_DAY
31-AUG-21

SQL> SELECT last_DAY ('12-FEB-17') FROM DUAL;
output LAST_DAY
28-FEB-17

6) months_between
SQL> select months_between ('31-dec-81', hiredate) months_between from emp where hiredate between '1-jan-81' and '31-dec-81'
output MONTHS_BETWEEN

1.4516129
7.96774194
6.70967742
8.93548387
.903225806
10.3548387
10.2903226
3.09677419
3.7419348
.903225806

10 rows selected

c] Enhancement to the GROUP BY clause

Cube & Rollup function

Rollup enable on SQL statement the calculate multiple levels of subtotal across a specified group of dimensions. It also calculate a grand total

CUBE enables a SELECT statement to calculate subtotals for all possible combination of a group of dimension It also calculate a grand total. This is the set of information typically of for all across-tabular reports, so CUBE can calculate a cross-tabular report with a single select statement

```
SQL> SELECT
    deptno,
    job,
    count(*),
    sum(sal)
FROM
    emp
GROUP BY
    ROLLUP (deptno, job);
```

output:-

DEPTNO	JOB	COUNT(*)	SUM(SAL)
10	CLERK	1	1300
10	MANAGER	1	2450
10	PRESIDENT	1	5000
10		3	8750
20	CLERK	2	1900
20	ANALYST	2	6000
20	MANAGR	1	2975
20		5	10875

30	CLERK	1	950
30	MANAGER	1	2850
30	SALESMAN	4	5600
30		6	9400
		14	29025

13 rows selected

```
SQL> SELECT deptno, job, count(*), sum(sal)
  FROM emp
  GROUP BY CUBE (deptno, job);
```

DEPTNO	JOB	COUNT(*)	SUM(SAL)
		14	29825
	CLERK	4	4150
	ANALYST	2	6000
	MANAGER	3	8275
	SALESMAN	4	5600
	PRESIDENT	1	5000
10		3	8750
10	CLERK	1	1300
10	MANAGER	1	2450
10	PRESIDENT	1	5000
20		5	10875
20	CLERK	2	1900
20	ANALYST	2	6000
20	MANAGER	1	2975
30		6	9400
30	CLERK	1	950
30	MANAGER	1	2850
30	SALESMAN	4	5600

18 rows selected

Sr.no	Date	Topic
7	11-8-21	SQL- Subquery- basics

Subquery basics: A query inside a query (main query)

Create table empt and dept tables

```
SQL>create table empt
 (empno number(10),
  ename varchar(10),
  job varchar(15),
  sal number(10)
  deptno number(15));
Table created
```

```
SQL>create table deptt
 (deptno number(10),
  dname varchar(10),
  loc varchar(15));
```

Table created

2) Insert proper or valid value in both the table

```
SQL> insert into empt values(1,'smith','manager',2550,10)
1 row selected created
```

```
SQL> insert into empt values (2,'Rohan khilan','clerk',2850,20);
1 row created
```

```
SQL> insert into empt values(3,'Kunar','clerk',5000,50);
1row created
```

Amruta Shinde
21944S

GoodLuck | Page No.
Date

(50)

SQL> insert into empt values (4, 'pooja', 'manager', 12000, 10);
1 row ~~selected~~ created

SQL> insert into emp values (5, 'neha', 'salesman', 2000, 20);
1 row created

SQL> insert into empt values (6, 'mahini', 'Manager',
10000, 40);

1 row created

Now inserting values dept t tables
SQL> insert into dept t values (50, 'analyst', 'pune');
1 row created

SQL> select * from empt;

EMPNO	ENAME	JOB	SAL	DEPTNO
1	Smith	manager	2550	10
2	khilan	clerk	2850	20
3	Kunal	clerk	5000	50
4	pooja	manager	12000	10
5	neha	Salesman	20000	20
6	mahini	manager	10000	40

6 rows selected
Now inserting values deptt tables
SQL> insert into deptt values (50, 'analyst', pune');

1 row created

Amruta Shinde
219445

GoodLuck | Page No.
Date

(51)

In cohich dept po department pooja work<

SQL>insert into deptt values (60,'developer', 'chennai');

1 row created

SQL> insert into deptt values (10,'sales', mumbai');

1 row created

SQL> insert into deptt values (20,'marketing', 'delhi');

1 row created

SQL> insert into deptt values (40,'HR', 'banglore');

1 row created

SQL> insert into deptt values(30,'PR','chennai');

1 row created

SQL> select * from deptt;

DEPTNO	DNAME	LOC
50	analyst	pune
60	developer	chennai
10	sales	mumbai
20	marketing	delhi
40	HR	banglore
30	PR	chennai

6 rows selected

GoodLuck | Page No.
Date

5 In which department (dname) employee pooja work?

```
SQL> select dname from deptt
     where deptno=
                 (select deptno
                  from empt
                  where ename='pooja');
```

DNAME
Sales

6. How many employees work in mumbai?

```
SQL> select count(*) from empt
     where deptno =
                 (select deptno
                  from deptt
                  where loc = 'mumbai');
```

COUNT(*)
2

7 In which location does employee smith works?

```
SQL> select loc from deptt
     where deptno =
                 (select deptno
                  from empt
                  where ename = 'smith');
```

loc
mumbai

8. How many employees in mumbai are appointed as manager?

```
SQL> select count(*) from deptempt
     where deptno =
                 (select deptno
```

```
from deptt
where loc='mumbai' and job='manager');
```

COUNT(*)
2

9 In which location does clerk works?

```
SQL> select loc from deptt
 where deptno IN (select deptno
                  from empt
                  where job='clerk');
```

loc
delhi
pune

10 who gets the highest salary?

```
SQL> select ename from empt
 where sal = (select max(sal)
              from empt);
```

ENAME
neha

11. which employee earns more salary than employee "smith"?

```
SQL> select ename from empt
  where sal > (select sal
               from empt
               where ename='smith');
```

ENAME
Khilan
Kunal
pooja
neha
mohini

Nested sub-query - A query within a subquery

1) who gets second higher salary?

```
SQL> select ename from empt
     where sal IN (select max(sal)
                   from empt
                   where sal < (select max(sal)
                                from empt));
```

ENAME
Pooja

2) How many employee in the HR department earns in the range 5000 and 15000?

```
SQL> select count(*) from empt
     where deptno = (select deptno
                     from deptt
                     where dname = "HR and sal between
                                    5000 and 15000);
```

COUNT(*)
1

3) which job category has the most number of employees earning above 8000.

```
SQL> select job from empt
     where sal > 8000
     group by job
     having count(*) = (select max(count(*))
                        from empt
                        where sal > 8000
                        group by job);
```

JOB
Manager

Amruta Shinde
219445

GoodLuck | Page No.
Date

4. In which department has more no of employee

```
SQL> select dname from deptt
     where deptno IN(select deptno
          from empt
          group by deptno
          having count(*)= (select
               (select max(count(*))
               from empt
               group by deptno));
```

DNAME
marketing
Sales

Amruta Shinde
219445

GoodLuck | Page No.
Date

Sr no	Date	Topic
8	12-8-21	Practical 8

A] Types of subquery

1) Single row
2) Multiple row

Use of group function in a subquery

1) Single row subquery:-

```
SQL> select ename, JOB
     from emp
     where sal = (select min(sal)
                  from emp);
```

ENAME	JOB
SMITH	CLERK

2) Use of Having clause with subquery example:-

```
SQL> select deptno, min(sal)
     from emp
     group by deptno
     having min(sal) > (select min(sal)
                        from emp
                        where deptno = 20);
```

DEPTNO	MIN(SAL)
30	950
10	1300

3 Using any operator:-

```
SQL> select empno, ename from emp
     where sal<ANY(select sal
                   from emp
                   where JOB='MANAGER');
```

EMPNO	ENAME
7369	SMITH
7900	JAMES
7876	ADAMS
7521	WARD
7654	MARTIN
7934	MILLER

10 rows selected

Consider emp table:-

1) Subquery Comparison test (=, <, >, >=, <=):-

```
SQL> select ename
     from emp
     where sal>(select sal from emp
                where empno=7566);
```

ENAME
SCOTT
KING
FORD

```
SQL> select ename, sal, deptno
     from emp
     where JOB=(select JOB from emp
                where empno=7369);
```

ENAME	SAL	DEPTNO
SMITH	800	20
ADAMS	1100	20
JAMES	950	30
MILLER	1300	10

2) Subquery set membership test (IN):-

```
SQL> select ename, sal, deptno
     from emp
     where sal IN (select min(sal)
                   from emp
                   group by deptno);
```

NAME	SAL	DEPTNO
SMITH	800	20
JAMES	950	30
MILLER	1300	10

3) Use of ANY Operator:-

```
SQL> select ename, sal, job, empno
     from emp
     where sal < ANY (select sal
                      from emp
                      where job='MANAGER');
```

NAME	SAL	JOB	EMPNO
SMITH	800	CLERK	7369
JAMES	950	CLERK	7900
AMS	1100	CLERK	7876
RD	1250	SALESMAN	7521
MARTIN	1250	SALESMAN	7654
MILLER	1300	CLERK	7934

GARNER	1500	SALESMAN	7844
LEN	1600	SALESMAN	7499
CLARK	2450	MANAGER	7782
BLAKE	2850	MANAGER	7698

4 Use of ALL Operator:

```
SQL> select empno, ename, job
     from emp
     where sal > ALL (select avg(sal)
                      from emp
                      group by deptno);
```

EMPNO	ENAME	JOB
7566	JONES	MANAGER
7788	SCOTT	ANALYST
7902	FORD	ANALYST
7839	KING	PRESIDENT

B] Aggregating data using grouping function:-

1) Group by clause:

```
SQL> select deptno, min(sal), max(sal), count(*)
     from emp
     group by deptno;
```

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	950	2850	6
20	800	3000	5
10	1300	5000	3

2 Use of where clause in aggregating data by using group function:-

```
SQL> select deptno, min(sal), max(sal), count(*)
     from emp
     where sal > 1000
     group by deptno;
```

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	1250	2850	5
20	1100	3000	4
10	1300	5000	3

3 Use of like operator in aggregating data by using group function:-

```
SQL> select deptno, min(sal), max(sal), count(*)
     from emp
     where ename like 'J%'
     group by deptno;
```

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	950	950	1
20	2975	2975	1

4) Use of order by clause in aggregating data by using group function:

```
SQL> select deptno, min(sal), max(sal), count(*)
     from emp
     where ename like 'J%'
     group by deptno
     order by deptno;
```

GoodLuck | Page No.
Date

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
20	2975	2975	1
30	950	950	1

5 use of having clause in aggregating data by using group function:.

```
SQL> select deptno, min(sal), max(sal), count(*)
     from emp
     where sal<1000
     group by deptno
     having min(sal)<1000;
```

DEPTNO	MIN(SAL)	MAX(SAL)	COUNT(*)
30	950	950	1
20	800	800	1

Amruta Shinde
219445

Page No.
Date

62

Sl.no	Date	Topic :-
9	2-9-21	Subquery & joins

Subquery & Joins

1 List the details of employees whose salary is more than the employee name BLAKE

```
SQL> select * from emp
     where sal >(select sal from emp
                 where ename = 'BLAKE');
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7566	Jones	Manager	7839	03-APR-81	2975		20
7788	Scott	Analyst	7566	19-APR-87	3000		20
7839	King	President		17-NOV-81	5000		10
7902	Ford	Analyst	7566	03-DEC-81	3000		20

2 List the employee who's sal is same as FORD or SMITH is desc order of sal

```
SQL>select ename from emp
    where sal IN(select sal from emp
                 where ename='FORD'
                 or ename = 'SMITH')
                 order by sal desc;
```

ENAME
FORD
Scott
Smith

Amruta Shinde
219495

3 Find details of highest paid employees.

```
SQL> select ename from emp
     where sal = (select max(sal)
                  from emp);
```

ENAME
King

4. Find highest paid employees of sales department

```
SQL> select ename, sal from emp
     where sal = (select max(sal)
                  from emp
                  group by deptno
                  having deptno = (select deptno
                                   where dname = 'Sales'))
```

ENAME	sal
BLAKE	2850

5 List the name of department with highest number of employees.

```
SQL> select dname from dept
     where deptno = (select deptno from emp
                     group by deptno
                     having count(*) =
                                (select max (count(*))
                                 from emp
                                 group by deptno));
```

DNAME
Sales

Amruta Shinde
219445

Page No.
Date

6. list the name of the employees who are getting highest salary department wise

- SQL> select ename from emp
  where sal IN (select max (sal)
  from emp
  group by deptno);

ENAME
Blake
Scott
King
Ford

7. list the names of the department where more than average number of employees are working

SQL > select dename from dept
where deptno IN (select deptno from emp
group by deptno
having count (*) >
(select avg (count (*))
from emp
group by deptno));

DNAME
Sales

Page No.
Date

8 List the details of most recently hired employee where deptno = 30

```
SQL> select ename from emp
     where hiredate = (select max(hiredate)
                       from emp
                       where deptno = 30)
```

ENAME
James
Ford

Amruta shinde
219445

GoodLuck | Page No.
Date

(66)

Sr no	Date	Topic
10	13-8-21	Permission

How to create user, How to revoke and grand permission from him or her

Step 1:- root login
SQL> conn
Enter user-name: sys as sysdba
Enter password: press enter-key
connected

Step 2:- Creating user
SQL> create user Amruta identified by 123;
user created

Step 3:- Granting permission to create table:-
SQL> grant create table to Amruta;
Grant succeeded.

~~SQL> grant~~
Step 4:-
SQL> grant create session to Amruta;
Grant Succeeded

Now login with your created user
SQL> conn
user-name: Amruta
Enter password
connected

GoodLuck | Page No.
Date / /

SQL> create table c(a b number);
Table created

Now again do root login
SQL> conn
Enter user-name: sys as sysdba
Enter password:
Connected

SQL> revoke create table from Amruta;
Revoke succeeded

SQL> revoke unlimited tablespace from Amruta;
Revoke succeeded

Again login with yours user-name:
SQL> conn
Enter user-name: Amruta
Enter password:
connected

SQL> create table b
(cd number(10));
create table b
*
ERROR at line 1:
ORA-01031: insufficient privileges

Root login:

```
SQL> conn
Enter user-name: sys as sysdba
Enter password:
connected
SQL> grant all privileges to Amruta;
Grant Succeeded

SQL> conn
Enter user-name: Amruta
Enter password:
connected
SQL> create table stud
    (sname varchar(10),
     sroll no number(10));
```

Amruta Shinde
219445

Sr.no	Date	Topic
11	26-8-21	clause

Clauses
1] from clause
2] where clause
3] Group by clause
4] ON clause
5] order clause
6] Having clause

1] Create a table book with following attribute (bookno, bname, authors, price

```
SQL> create table book
     (bookno  number(10),
      bname   varchar(20),
      author  varchar(20),
      price   number(10));
Table created
```

2] Find the details book of table
→ SQL> select * from book

backno	bname	Author	Price
1	Discovery of Indiq	J.L.Nehru	125
2	Descent of Man	Charle's Darvin	198
3	life Divine	Aurobindo	200
4	Human knowledge	B.Russel	225
5	Shakuntala	Kalidas	200
6	Gitanjali	R.N.Tigore	800

3. Per book name what is the maximum book price

```
SQL> select max(price)
     from book
     group by bname;
```

output

max price
125
198
200
225
200
300

4. Per book name how many books have anonymous(null) author

```
SQL> select count(*)
     from book
     where author = 'Null'
     group by bname;
```

output no rows selected

5. Per author how many books are priced between 100 and 200 including

```
SQL> select count(*)
     from book
     where price between 100 and 200;
```

output

COUNT(*)
4

6. How many different authors are listed in the table

→ SQL> ~~Select count (distin~~

SQL> select count (DISTINCT author)
from book;

COUNT(DISTINCTAUTHOR)
6

7 Per author how many books are written

SQL> select count(*)
from book
group by bname

O/P

COUNT(*)
1
1
1
1
1
1

Amruta Shinde
219495

Sr no	Date	Topic
12	27-8-21	SQL-statements-basic

1) Selecting all columns

SQL> select * from emp;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30

Selecting specific columns

SQL> select empno, ename, Deptno, sal from emp;

EMPNO	ENAME	DEPTNO	SAL
7369	SMITH	20	800
7499	ALLEN	30	1600
7521	WARD	30	1250
7566	JONES	20	2975
7654	MARTIN	30	1250
7698	BLAKE	30	2850
7782	CLARK	10	2450
7788	SCOTT	20	3000
7839	KING	10	5000
7844	TURNER	30	3100
7876	ADAMS	20	1100

2) selecting different Jobs for the employees
SQL> select DISTINCT JOB from emp;

JOB
CLERK
SALESMAN
PRESIDENT
MANAGER
ANALYST

3. selecting salary of employee under the heading
SQL> select sal as employee_salary from emp;

EMPLOYEE_SALARY
800
1600
1250
2975
1250
2850
2450
3000
5000
8100
1100

4. Using Arithmetic operators:
SQL> select empno, ename, sal, sal+200 as new_sal from emp;

EMPNO	ENAME	SAL	NEW_SAL
7369	SMITH	800	1000
7499	ALLEN	1600	1800
7521	WARD	1250	1450
7566	JONES	2975	3175
7654	MARTIN	1250	1450

14 rows selected

5 Concatination operator:

```
SQL>select('employee'||' '||ename||' '||'is working as'||job)
    from emp;
```

```
('EMPLOYEE'||''||ENAME||''||'ISWORKINGAS'||J
employee SMITH is working as CLERK
employee ALLEN is working as SALESMAN
employee WARD is working as SALESMAN
employee JONES is working as MANAGER
employee MARTIN is working as SALESMAN
employee BLAKE is working as MANAGER
employee CLARK is working as MANAGER
employee SCOTT is working as ANALYST
employee KING is working as PRESIDENT
employee TURNER is working as SALESMAN
employee ADAMS is working as CLERK
```

6. select all the details of employee whose salary is greater than 2000.

```
SQL> select * from emp
     where sal > 2000;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	3100	0	30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20

7 rows selected

7. select the name of employee working as analyst

```
SQL> select ename from emp where JOB = 'Analyst';
```

ENAME
SCOTT
FORD

8 select the employees name and deptno and salary who are working in either department 20 or as salesman

```
SQL> select ename, deptno, job from emp where deptno = 20 or
JOB = 'salesman';
```

ENAME	DEPTNO	JOB
SMITH	20	CLERK
JONES	20	MANAGER
SCOTT	20	ANALYST
ADAMS	20	CLERK
FORD	20	ANALYST

9 select the employee name, deptno, salary who are working either in department no. 10 with sal >1200
SQL> select ename, deptno, sal from emp where sal>1200 AND deptno = 10;

ENAME	DEPTNO	SAL
CLARK	10	2450
KING	10	5000
MILLER	10	1300

10. Display the details of employees where they are working as salesman, manager and analyst
SQL> select * from emp where JOB IN ('salesman', 'manager', analyst');

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20

11. Display the details whose JOB starts with A.
SQL> select * from emp where JOB like 'A%'

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7888	SCOTT	ANALYST	7566	19-APR-87	3000		20
7902	FORD	ANALYST	7566	03-DEC-81	3000		20

12 Display the name of employees when the Name of the employee ends with 'A'
SQL> select ename from emp where ename like '%A';

ENAME
POOJA

13 Display the name and department number of employee is exac of 5 characters
SQL> select ename, deptno from emp where ename like'__

ENAME	DEPTNO
SMITH	20
ALLEN	30
JONES	20
BLAKE	30
CLARK	10
SCOTT	20
ADAMS	20
JAMES	30
POOJA	20

14 Display the details where when comm is null:
SQL> select * from emp where comm=0;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7844	TURNER	SALESMAN	7698	08-SEP-81	3100	0	30

15 Display the details of employees with respect to ascending order of their salaries Display ename and salary.
SQL> select ename, sal from emp order by sal;

ENAME	SAL
SMITH	800
DAMES	950
ADAMS	1100
WARD	1250
MARTIN	1250
MILLER	1300
ALLEN	1600

Amruta shinde
219445

classmate
Date
Page

CLARK	2450
BLAKE	2850
JONES	2975
SCOTT	3000

16 Display number of employees working for each designation
select job, count(*) from emp group by job;

JOB	COUNT(*)
CLERK	4
SALESMAN	4
PRESIDENT	1
MANAGER	4
ANALYST	2

17 Display total salary of employees working for each designation
SQL>select JOB, SUM(sal) as total_sal from emp group by job;

JOB	TOTAL-SAL
CLERK	4150
SALESMAN	7200
PRESIDENT	5000
MANAGER	12275
ANALYST	6000

18 Display the total salary by grouping jobs and department number when count salary is greater than 2
SQL> select job, deptno, sum(sal) from emp group by job, deptno having count(sal)>2;

JOB	DEPTNO	SUM(SAL)
SALESMAN	30	7200

Amruta Shinde
219445

79

Sr.no	Date	Topic
13	2-9-2021	Grouping Function

Grouping Function is also known as aggregate Function since they have an ability to process entire table & obtain single value

1] create table employee with attribute (deptno, empno, ename, job & sal);

```
SQL> create table employee,
     (deptno number(10)
      empno number(10),
      ename varchar(20),
      Job   varchar(20),
      Sal   number(10));
Table created
```

SQL> select * fr

2 Find All detail from employee table

```
SQL> select * from employe;
```

deptno	empno	ename	Job	~~sal~~ sal
10	7369	Pooja	CLERK	200
30	7499	Priti	CLERK	500
10	7521	Rahul	Salesman	1200
20	7566	Sachin	Manager	1400
20	7565	Pradeep	salesman	500
30	7654	Poonam	President	10000
20	7782	shweta	clerk	200
10	7839	Reetu	HR	6000
30	7844	Rishi	Analyst	7500
~~30~~ 10	7902	Hello	Manager	5000

3 what is the maximum salary in employee table

SQL> select max(sal) from employee.

Max(sal)
10000

4 what is max salary from all clerk

SQL> select max(sal)
from employee where Job = 'CLERK'
.

MAX(SAL)
500

5 what is the average salary for emp of deptno 10

SQL> select avg(sal)
from employee
where deptno = 10;

AVG(SAL)
3100

6 what is the minimum salary in employee table other than department 10, 20

SQL> select min(sal)
from employee
where deplo NOT IN (10, 20);

MIN(SAL)
500

7 How many employee has first and last alphabet as P

```
SQL> select count(*)
     from employee
     where ename like '%P' and ename like 'P%';
```

count(*)
1

8 How many manager are listed in the table

```
SQL> select count(*)
     from employee
     where job='manage';
```

Count(*)
2

9 which are the different Jobs specified in the table

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

JOB
HR
CLERK
MANAGER
SALESMAN
PRESIDENT
ANALYST

10 How many jobs are their in employee table

```
SQL> select count(Distinct job)
     From employee;
```

Count(Distinct job)
6

Amruta Shinde
219445

Page No.
Date

Sr no	Date	Topic
14	7-9-21	Triggers

Steps:
SQL> conn
Enter user-name: scott
Enter password: tiger
Connected

Creating Database Triggers:-
1) Create a trigger to insert the name of the student in UPPERCASE

```
SQL> create table stud
       (rno int,
        sname varchar(10));
```

Table created

```
SQL> create or replace tigger ucase
     before insert on stud
     for each row
     when (new.sname != upper(new.sname))
     begin
     :new.sname := upper(:new.sname);
     /
```

Trigger created

Amruta Shinde
219445

SQL > insert into stud values(1,'james');

1 row created

SQL > insert into stud values(2,'jyoti');

1 row created

SQL> select * from stud;

RNO	SNAME
1	JAMES
2	JYOTI

Amruta Shinde
219445

Page No.
Date

(85)

Q.2 Create a trigger to update the marks of a student by 20 before inserting a record in a table

```
SQL>create table t1(rno int, marks int);
Table created

SQL> create or replace trigger beforeupdate
     before insert on t1
     for each row
     begin
     :new.marks:=:new.marks+20;
      end;
      /
Trigger created

SQL> insert into t1 values(1,10);
1 row created

SQL>insert into t1 values(2,20);
1 row created

SQL> select * from t1;
```

RNO	MARKS
1	30
2	40

Amruta Shinde
219445

Q3 Create a trigger to insert the username and login date in a table
Note: For this practical do root login

login with : sys as sysdba
Password:- Press enter-key

```
SQL> create table login
     (un varchar(10),
      logindate date);
```

Table created

```
SQL> create or replace trigger tlogin
     after logon on database
     begin
     insert into login values (user, sysdate);
     end;
     /
```

Trigger created

```
SQL> select * from login;
```

UN	LOGINDATE
SYSMAN	06-AUG-19

4 Create a trigger to insert delete records in a table

```
SQL> Create table empdup as select * from emp;
Table created

SQL> create table empdelete
     (empno number, sal number(7,2));

Table created

SQL> Create or replace trigger trgdlt after
     delete on empdup for each row
     begin
     insert into empdelete values(:old.empno,:old.sal);
     end trgdlt;
     /

Trigger created

SQL> delete from empdup where empdno=7902;
1 row deleted

SQL> select * from empdelete;

   EMPNO        SAL
    7902        3000

SQL> delete from empdup where empno=7900;
1 row deleted

SQL> select * from empdelete;
   EMPNO        SAL
    7902        3000
    7900         950
```

Amruta shinde
219445

(88)

Page No.
Date

Sr no	Date	Topic
15	7-9-21	

Topic:-
Finding out 1st, 2nd, 3rd, 4th, 5th. . . . Nth from highest Salary from employee table

1 Creating employee table with attribute employee_id

```
CREATE TABLE employees
(employee_id INT PRIMARY KEY,
 salary INT
);
```

Table created

2 Insert values into employees tables:

```
SQL> insert into employees values (1, 100000)
1 row created
SQL> insert into employees values (2,20000);
1 row created
SQL> insert into employees values(3,20000);
1 row created
SQL> insert into employees values(4, 30000);
1 row created
SQL> insert into employees values (5,40000);
1 row created
SQL> insert into employee values (6,50000);
1 row created
```

Amruta Shinde
21944S

3 Find out details of employee table.
SQL> select * from employees;

EMPLOYEE ID	SALARY
1	10000
2	20000
3	20000
4	30000
5	40000
6	50000

4 Find out highest ~~from~~ salary from employee table
SQL> select max(salary)
from employees;

MAX(SALARY)
50000

5 Find out second highest salary from employee table
SQL> select max(salary)
from employee
where salary Not In (select Max(salary)
from employee);

MAX(SALARY)
4000

Amruta Shinde
219445

Page No.
Date

90

Note: whenever we want find out Nth highest salary from the same table, we create alies of table for doing and finding out count of distinct salary & their by displaying the result.

Syntax:-

```
select employee_id, salary
from Employees e1
where N-1 = (select count(Distinct salary)
             from employee e2
             where e2.salary > e1.salary);
```

~~Find out thid third higest~~

6 Find out third highest salary from employee table

```
SQL> select employee_id, salary
     from Employees e1
     where 2 = (select count(Distinct salary)
                from employees e2
                where e2.salary > e1.salary
```

EMPLOYEE_ID	SALARY
4	30000

7 Find Fourth highest salary from employee table

```
SQL> select employee_id, salary
     from Employees e1
     where 3 = (select count(DISTINCT salary)
                from employees e2
                where e2.salary > e1.salary);
```

EMPLOYEE_ID	SALARY
2	20000
3	20000

8. Find out 5th highest salary from employee table

```
SQL> select employee_id, salary
     from Employees e1.
     where 4 = (select count(DISTINCT salary)
                from employees e2
                where e2.salary > e1.salary
```

EMPLOYEE_ID	SALARY
1	10000

Amruta Shinde
219445

Sr no Date Topic
16 15-09-21 PLSQL Basic

PL/SQL BASICS

a] Declaring Variables:-

1 write PL/SQL block to find and display simple interest by accepting the input of principle amount, rate of interest & number of terms from user

```
DECLARE
p number:=&Principle_Amount;
n number:=&Rate_of_Interest;
r number:=&Number_of_Terms;
si number;
BEGIN
si:=(p*n*r)/100;
dbms_output.put_line('Simple Interest="||si);
END;
/
```

```
Enter value for principle_amount: 100
old 2: p number:=&Principle_Amount;
new 2: p number:=100;
Enter value for rate_of_Interest: 100
old 8: n number:=&Rate_of_Interest;
new 3: n number:=100;
Enter value for number_of_terms: 10
old 4: r number:=&Number_of_Terms;
new 4: r number:=10;
Simple Interest=1000
PL/SQL procedure successfully completed
```

2 Write a PL/SQL block to display the total number of employees from EMP table

```
DECLARE
n number
BEGIN
select count(*) into n from emp;
dbms output.put_lines ('Total Numbers of Employees =' ||n);
END;
/
```

Total Numbers of Employees = 14
PL/SQL procedure successfully completed

3 write a PL/SQL block to commit the changes in the table only when revise salary is greater than 1000 otherwise undo the changes using save point. The salary can be updated by 400 of the employees whose employee number is given by the user

```
DECLARE
no number := &empno;
s number;
n number;
BEGIN
SELECT sal into s from emp
where empno = no;
savepoint p1;
update emp
set sal = sal + 400
where empno = no;
```

Amruta Shinde
219445

Page No.
Date

94

```
SELECT sal into n from emp
where empno=no;
if n>1000 then
commit;
else
rollback to p1;
end if;
dbms_output.put_line('original salary = '||s);
dbms_output.put_line('Revised salary = '||n);
END;
/
```

```
Enter value for empno: 7900
old 2: no number:=&empno;
new 2: no number:= 7900;
Original salary = 950
Revised salary = 1350
PL/SQL procedure successfully completed.
```

Note: Agar salary phale se 950 and usme hum 400 Plus krenge then 1350 Hoga so wo Jada Hoga 1000 se output display hoga & main table Me (EMP) wo changes dikhata hai. i.e 1350 hota hai EMP Table

Amruta Shinde
219445

Page No.
Date

(95)

```
DECLARE
no number:=&empno;
s number;
n number;
BEGIN
SELECT sal into s from emp
where empno=no;
savepoint p1;
Update emp
set sal=sal+100
where empno=no;
SELECT sal into n from emp
where empno=no;
if n> 1000 then
commit;
else
rollback to p1;
end if;
dbms_output.put_line('original salary' ||s);
dbms_output.put_line('Revised salary' ||n);
END;
/
```

```
Enter value for empno :7900
old 2: no number:=&empno;
new 2: no number:=7869;
Original salary = 800
Revised salary = 900

PL/SQL procedure successfully completed
```

Amruta Shinde
219445
96

b) Writing Control structures:-

1) Write a PL/SQL Block to display numbers from 1-10 using exit statement

```
DECLARE
i number:= 1;
BEGIN
loop
dbms_output.put_line(i);
i= i+1;
if i = 11 then
exit;
end if;
end loop;
END;
/
```

```
1
2
3
4
5
6
7
8
9
10
PL/SQL procedure successfully completed
```

Amruta Shinde
219445

Page No. 97
Date

write a PL/SQL block to get the details of table marks (roll no, marks1, marks2, grade) out of 100 for marks1 & mark2, grade) out of 100 for marks1 & marks2 Display & update grade in table marks with if statement

```
Create table marks
(roll_no number,
marks1 number,
marks2 number,
grade char);
```

Table created

```
INSERT INTO MARKS VALUES(120,67,57,NULL);
1 ROW CREATED

SQL> select * from marks;

ROLL_NO   MARKS1   MARKS2 G
  120       67        57

DECLARE
total constant number:=200;
percent number;
m1 number;
m2 number;
m number;
BEGIN
select marks1, marks2 into m1, m2 from marks;
m:=m1+m2;
percent:=(m/total)*100;
```

Amruta shinde
219995
98

```
if percent>70 then
update marks
set grade='A';
elsif percent>60 and percent<70 then
update marks
set grade='B';
else
update marks
set grade='C';
end if;
END;
/
```

Output PL/SQL procedure successfully completed

```
SQL> select * from marks;
ROLL_NO    MARKS1    MARKS2 G
    120        67        57 B
```

Write a PL/SQL block to accept job from EMP table
Given the following raise in the salary:
1. By 9% if job is clerk
2. By 8% if job is manager
3. By 7% if job is salesman
Update the salary of EMP table

```
DECLARE
j varchar(20)=upper('&Job');
BEGIN
if j='CLERK' then
update emp
```

```
set sal = sal+(sal*0.09)
where job=j;
elsif j = 'MANAGER' then
Update emp
set sal=sal+(sal*0.08)
where job=j;
elsif j = 'SALESMAN' then
update emp
set sal = sal+(sal*0.07)
where job=j;
else
dbms_output.put_line("Invalid Entry");
END IF;
END;
/
```

output

```
Enter value for job: manager
old 2: j varchar(20):= upper('&Job');
new 2: j varchar(20):= upper('manager');

PL/SQL procedure successfully completed
SQL> select job, sal from emp;

JOB          SAL
CLERK        800
SALESMAN     1600
SALESMAN     1250
MANAGER      3213
SALESMAN     1250

14 rows selected
```

Amruta Shinde
219445

Page No.
Date

100

Write a PL/SQL block to check whether the number entered by the user is even or odd

```
DECLARE
n number:=&num;
BEGIN
if(n mod 2)=0 then
dbms_output.put_line(n||' is Even Number');
else
dbms_output.put_line(n||' is odd Number');
end if;
END;
/
```

output
```
Enter value for num: 6
old 2: n number:=&num;
new 2: n number:= 6;
6 is Even Number

PL/SQL procedure successfully completed
```

6 Write a PL/SQL block to calculate the area of circle till radius greater than 10. Also insert the radius & area in table aoc.

```
CREATE TABLE AOC
(radius number,
area number);

Table created
```

Amruta Shinde
219445

101

```
DECLARE
r number:=1;
pi constant number:=3.14;
a number;
BEGIN
while(r<=10)
loop
a:=pi*r*r;
insert into aac values(r,a);
r:=r+1;
end loop;
END;
/
```

Output PL/SQL procedure successfully completed

```
SQL> select * from aac;
```

RADIUS	AREA
1	3.14
2	12.56
3	28.26
4	50.24
5	78.5
6	113.04
7	153.86
8	200.96
9	254.34
10	314

10 rows selected

Amruta Shinde
219445
Page No.
Date
(102)

Write a PL/SQL block using goto & null statement to update the salary of an employee when minimum salary is greater than 1000.

```
DECLARE
s number;
BEGIN
select min(sal) into s from emp;
if s>1000 then
update emp
set sal = sal+500;
goto mylabel;
else
   null;
end if
<<mylabel>>
dbms_output.put_line("salary updated");
END;
/
```

Output- Salary updated

PL/SQL procedure successfully completed

write a PL/SQL block to book a ticket for a movie. The tickets are of two type's deluxe rows(D) & ordinary rows(O) while booking the ticket the customer may ask 'D' or 'O' and number of ticket for deluxe the rate is 350 & for ordinary 200. Find the total amount that the customer will pay & number of tickets (using case statements)

Amruta Shinde
219445
103

```
DECLARE
ticket char := Upper('&Type_of_Ticket');
t number := &Number_of_Ticket;
sum1 number := 0;
drate number := 350;
orate number := 200;
BEGIN
case
when ticket = 'D' then
sum1 := t* drate;
dbms_output.put_line(t||' of Delux ticket will cost you'
    ||sum1);
when ticket = 'O' then
 sum1 := t* orate
dbms_output.put_line(t||' of ordinary ticket will cost you '||sum1);
else
dbms_output.put_line('Invalid Entry');
end case;
END;
/
```

output 1: Enter value of Type_of_Ticket: O
old 2: ticket char := upper ('&Type_of_Ticket');
new 2: ticket char := Upper ('O');
Enter value for number_of_ticket: 5
old 3: t number := &Number_of_Ticket;
new 3: t number := 5;
5 of ordinary ticket will cost you 1000

PL/SQL procedure successfully completed.

Amruta Shinde
219445
104

Page No.
Date

output 2:- Enter value for type_of_ticket:d
old 2: ticket char:= upper('&Type_of_Ticket');
new 2: ticket char:= upper('d');
Enter value for number_of_ticket:5
old 3: t number:=&Number_of_Ticket;
new 3: t number:=5;
5 of Delux ticket will cost you 1750

PL/SQL procedure successfully completed

Amruta Shinde
219445

Page No.
Date

105

Sr no	Date	Topic:
17	21-9-21 28-9-21	Composite Data types, cursors & Exceptions

a) Working with Composite Data Types

```
DECLARE
Type emprec IS RECORD
(ename varchar(50),
job varchar(50),
Sal number,
empno number);
el emprec;
BEGIN
select ename, job, sal, empno into el from emp
where empno=&empno;
dbms_output.put_line(el.ename||' '||el.job||' '||el.sal||' '||
                     el.empno);
END;
/
```

Output
```
Enter value for empno: 7902
old 10: where empno=&empno;
new 10: where empno = 7902;
FORD ANALYST 3000 7902

PL/SQL procedure successfully completed
```

Amruta shinde
219445
(106)

```
DECLARE
TYPE names_table IS TABLE OF VARCHAR2(10);
TYPE grades IS TABLE OF INTEGER;
names names_table;
marks grades;
total integer;
BEGIN
names:= names_table('Amaira', 'Pritam', 'Ayan', 'Amir');
marks:= grades(98,97,78, 87);
total:= names.count;
dbms_output.put_line('Total '|| total ||' students');
FOR i IN 1.., total LOOP
dbms_output.put_line('student:'||names(i)||', Marks:'||
                                                marks(i));

end loop;
END;
/
```

output
```
Total 4 students
student: Amaira, Marks: 98
student: Pritam, Marks: 97
student: Ayan, Marks: 78
student: Amir, Marks: 87

PL/SQL procedure successfully completed
```

Amruta Shinde
219445


```
DECLARE
vv_emp emp%.ROWTYPE
BEGIN
SELECT *
INTO vv_emp
FROM emp
WHERE empno = 7900;
/* to display each element of record, reference
 each attribute of record with dot notation */
DBMS_OUTPUT.PUT_LINE('Employee Details: '||vv_emp.
                                    empno||' '||
vv_emp.ename||' '||vv_emp.sal);
END;
/
```

output Employee Details: 7900 JAMES 1350

PL/SQL procedure successfully completed

B] Writing Explicit Cursors

1) write a PL/SQL block to DISPLAY the employee name along with their salary using WHILE LOOP

```
DECLARE
emp_rec EMP%.ROWTYPE;
CURSOR c IS
SELECT * FROM EMP;
BEGIN
OPEN c;
FETCH c INTO emp_rec;
```

```
WHILE(C%.FOUND)
  LOOP
  DBMS_OUTPUT.PUT_LINE(emp_rec.ename||''||emp_rec.sal);
  FETCH c INTO emp_rec;
  END LOOP;
CLOSE C;
END;
/
```

Output KING 5000
BLAKE 5000
CLARK 2850
JONES 2450
SCOTT 2975
FORD 3000
SMITH 3000
ALLEN 800
WARD
MARTIN
TURNER
ADAMS
JAMES
MILLER

PL/SQL procedure successfully completed

2 Write a PL/SQL block to check whether the CURSOR is OPEN or not and to DISPLAY the appropriate message using EXPLICIT CURSOR

```
DECLARE
CURSOR c IS
SELECT * FROM EMP;
 BEGIN
IF c%ISOPEN
   THEN
   DBMS_OUTPUT.PUT_LINE('CURSOR IS OPEN');
ELSE
   DBMS_OUTPUT.PUT_LINE('CURSOR IS NOT OPEN');
END IF;
END;
/
```

olp CURSOR IS NOT OPEN

PL/SQL procedure successfully completed

3. write a PL/SQL block to DISPLAY the employee name and their hire date using FOR LOOP

```
DECLARE
CURSOR c IS
SELECT * FROM EMP;
BEGIN
FOR emp_rec IN c
  LOOP
  DBMS_OUTPUT.PUT_LINE(emp_rec.ename||' '||emp_rec.
                                          hiredate);
  END LOOP;
END;
/
```

o/p
```
KING 17-NOV-81
BLAKE 01-MAY-81
CLARK 09-JUN-81
JONES 02-APR-81
SCOTT 19-APR-87
FORD 03-DEC-81
SMITH 17-DEC-80
ALLEN 20-FEB-81
TURNER 08-SEP-81
ADAMS 23-MAY-87
JAMES 03-DEC-81
MILLER 23-JAN-82

PL/SQL procedure successfully completed
```

4 Write a PL/SQL block to DISPLAY the employee name, along with the Jobs who are located in 'NEW YORK'.

```
DECLARE
CURSOR c IS
SELECT * FROM EMP
WHERE deptno=
(SELECT deptno FROM dept
WHERE loc='NEW YORK');
BEGIN
FOR emp_rec IN c
    LOOP
    DBMS_OUTPUT.PUT_LINE(emp_rec.ename||' '||emp_rec.job);
  END LOOP;
END;
/
```

Amruta Shinde
219445

Page No.
Date

111

O/P KING PRESIDENT
CLARK MANAGER
MILLER CLERK

5 Write a PL/SQL block to DISPLAY the employee name along with their salary who belongs to department number is 30

```
DECLARE
CURSOR s1(dno NUMBER)IS
SELECT * FROM EMP
WHERE deptno=dno;
d emp.deptno%TYPE:=&d;
BEGIN
FOR emp_rec IN s1(d)
  LOOP
  DBMS_OUTPUT.PUT_LINE(emp_rec.ename||' '||emp_rec.sal);
  END LOOP;
END;
/
```

O/P Enter value for d: 30
old 5: d emp.deptno%TYPE:=&d;
NEW 5: demp.deptno%TYPE = 30;
BLAKE 3078
ALLEN 1600
WARD 1250
MARTIN 1250
TURNER 1500
JAMES 1390

PL/SQL procedure successfully completed

6 Write a PL/SQL block to DISPLAY the name of the employee whose name starts with 'A' of department number is 20, where department number is passed by the user

```
DECLARE
CURSOR s1(dno NUMBER) IS
SELECT * FROM EMP
WHERE ename LIKE 'A%' AND deptno=dno;
d emp.deptno%TYPE:=&d
BEGIN
FOR emp_rec IN s1(d)
  LOOP
  DBMS_OUTPUT.PUT_LINE(emp_rec.ename)
  END LOOP;
END;
/
```

o/p
```
Enter value for d:20
old 5: d emp.deptno%.TYPE:=&d
new 5: d emp.deptno%.TYPE:=20;
ADAMS

PL/SQL procedure successfully completed
```

Amruta Shinde
219495
(113)

c] HANDLING EXCEPTIONS:-

1 Implement system defined exception

```
DECLARE
  dno emp.deptno%type := &deptno;
  en emp.ename%type;
  esal emp.sal%type;
BEGIN
  SELECT ename, sal INTO en, esal
  FROM emp
  WHERE deptno=dno;
  DBMS_OUTPUT.PUT_LINE(en|| ' '||esal);
EXCEPTION
  WHEN no_data_found THEN
  dbms_output.put_line('NO such employee exists!');
WHEN too_MANY_ROWS THEN
  dbms_output_line('cannot fetch more than one record!'
END;
/
```

O/P

```
Enter value for deptno: 10
old 2: dno emp.depto%type := &deptno;
new 2: dno emp.deptno%type := 10;
Cannot fetch more than one record!

PL/SQL procedure successfully completed
```

2. Implementing User defined exception

```
DECLARE
EN EMP.ENAME%TYPE;
eno emp.empno%type := &eno;
```

```
my_excep exception;          ... userdefined exception
BEGIN
IF eno<7900 then
raise my_excep;              ... exception raised
else
  SELECT ename INTO EN FROM EMP WHERE EMPNO=eno;
  DBMS_OUTPUT.PUT_LINE(en);
end if;
  EXCEPTION
  WHEN my_excep THEN
  DBMS_OUTPUT.PUT_LINE('empno must be greater than
                                        7900');
  END;
  /
```

olp
```
Enter value for eno: 6999
old 3: eno emp.empno%type:=&eno;
new 3: eno emp.empno%type:=6999;
empno must be greater than 7900

PL/SQL procedure successfully completed
```

3 write a PL/SQL block to implement raise_application_error() when the age of the student is entered less than 0

```
declare
a int:=&a;
age_exception;
begin
if a>0 then
```

```
dbms_output.put_line('age is'|| a);
else
raise age_ex;
end if;
exception
when age_ex then
raise_application_error(-20001,'age cannot be 0 or
                                less than zero');
end;
/
```

```
o/p  Enter the value for int: 10
     old 8: age_ex exception : age
     new 8: ag_ex exception = 10;
```

4) write a PLlsql block to raise an exception when an employee name does not start with 'A'

```
DECLARE
en varchar(10); = '4en';
name_ex exception;
BEGIN
if en like 'a%' then
   dbms_output.put_line('name is'||en);
else
  raise name_ex;
end if;
exception
when name_ex then
```

```
    dbms_output.put_line('name should start with
END;                                          A');
/
```

O/P
```
Enter value for en: james
old 2: en varchar(10):='&en';
new 2: en varchar(10)='james';
name should start with A

PL/SQL procedure successfully completed
```

5 write a PL/SQL block implements exceptional propogation

```
DECLARE
   div_ex exception;
   n1 number:=&n1;
   n2 number:&n2;
   n3 number:&n3;
   n4 number:=&n4;
   div1 number;
   div2 number;
BEGIN
   div1:=n1/n2;
   dbms_output.put_line('division is:'||div1);
   begin
   if n4<=0 then
     raise div_ex;
   else
     div2:=n3/n4;
     dbms_output.put_line('division is:'||div2);
```

Amruta Shinde
219445

(117)

PAGE No.
DATE / /

```
    end if;
END;
EXCEPTION
    when zero_divide then
    dbms_output.put_line('division with zero is not
                                        allowed');
    when div_ex then
    dbms_output.put_line('value of n4 should be greater
                                        than zero');
    END;
    /
```

o/p
```
Enter value for n1: 4
old 3: n1 number:=&n1;
new 3: n1 number=4;
Enter value for n2: 2
old 4: n2 number:=&n2;
new 4: n2 number:=2;
Enter value for n3: 12
old 5: n3 number:=&n3;
new 5: n3 number:=12;
Enter value for n4: 0
old 6: n4 number:&n4;
new 6: n4 number:=0;
division is::2
value of n4 should be greater than zero

PL/SQL procedure successfully completed
```

Amruta Shinde
219445

(118)

Sr.no	Date	Topic:-
18	4-10-21	PROCEDURES AND FUNCTIONS:-

a) Creating Procedures

1) Create an empty procedures, replace that procedu -re to print 'HELLO WORLD' & execute the procedur

```
CREATE PROCEDURE pdemo
IS
BEGIN
NULL;
END;
/
```

output Procedure created

```
CREATE OR REPLACE PROCEDURE pdemo
IS
BEGIN
DBMS_OUTPUT.PUT_LINE('HELLO WORLD');
END;
/
```

output Procedure created

```
EXEC pdemo;
```

Output:- HELLO WORLD

2 Create a stored procedure and display employee name by accepting employee number from user

```
CREATE OR REPLACE PROCEDURE pdemo (num IN NUMBER,
name OUT VARCHAR)
IS
BEGIN
SELECT ENAME INTO name FROM emp
WHERE EMPNO=num;
DBMS_OUTPUT.PUT_LINE('EMPLOYEE NAME = '||name);
END;
/
```

o/p ~~prod~~ Procedure created

```
DECLARE
num emp.empno%type:=&empno;
name emp.ename%type;
BEGIN
Pdemo(num,name);
END;
/
```

o/p Enter value for empno: 7900
old 2: num emp.empno%type:=&empno;
new 2: num emp.empno%type:=7900;
EMPLOYEE NAME = JAMES

PL/SQL procedure successfully completed

Amruta Shinde
219445
Page No.
Date
120

3 Create a procedure to Display the name & salary of the employee by using Cursor

```
CREATE OR REPLACE PROCEDURE pdemo
IS
CURSOR c IS
SELECT * FROM emp;
BEGIN
FOR cr IN c
LOOP
  DBMS_OUTPUT.PUT_LINE(cr.ename||' '||cr.sal);
END LOOP;
END;
/
```

o/p Procedure created

```
EXEC pdemo;
```

o/p KING 5000
BLAKE 3078
CLARK 2646
JONES 3213
SCOTT 3000
FORD 3000
SMITH 800
ALLEN 1600
WARD 1250
MARTIN 1250
TURNER 1500
ADAMS 1100
JAMES 1350
MILLER 1300
PL/SQL procedure successfully completed

4 Create a procedure to insert data into table Demo(roll no, name) from the procedure

```
CREATE TABLE DEMO(roll_no NUMBER, name VARCHAR(10));
```

o/p Table created

```
CREATE OR REPLACE PROCEDURE pdemo(num IN NUMBER,
name IN VARCHAR)
 IS
 BEGIN INTO demo VALUES(num, name);
 INSERT INTO demo VALUES(num, name);
 END;
/
```

o/p Procedure created

```
DECLARE
 num NUMBER:=&roll_no;
 name VARCHAR(10):='&name';
 begin
 pdemo(num, name);
 END;
/
```

o/p Enter value for roll_no: 5
old 2 num NUMBER:=&roll_no;
new 2: num NUMBER:=5;
Enter value for name: JOHN
old 3: name VARCHAR(10):='&name';
new 3: name VARCHAR(10):='JOHN';

PL/SQL procedure successfully completed

```
SELECT * FROM demo;
```

o/p

ROLL_NO	NAME
5	JOHN

5. Write a PL/SQL block to show the forward declaration of Procedures.

```
DECLARE
PROCEDURE p2; ..... FORWARD DECLARATION

PROCEDURE P1
AS
BEGIN
DBMS_OUTPUT.PUT_LINE('P1');
END;

PROCEDURE P2
AS
BEGIN
DBMS_OUTPUT.PUT_LINE('P2');
P1;
END

BEGIN
P2;
END;
/
```

o/p

```
P2
P1
```

B] Creating FUNCTIONS:-

1. Create a function to Display the current date

```
CREATE OR REPLACE FUNCTION fdemo
return date
as
en date;
BEGIN
select sysdate into en from dual;
return en;
END;
/
```

o/p function created

```
**execute dbms_output.put_line(fdemo);
```

o/p 29-AUG-17
PL/SQL procedure successfully completed

```
DECLARE
d date;
BEGIN
d:=fdemo;
dbms_output.put_line(fdemo);
END;
/
```

o/p 29-AUG-17

PL/SQL procedure successfully completed

Amruta Shinde
219445

2 Create a function to find Addition of two numbers

```
CREATE OR REPLACE FUNCTION fdemo(a number, b number)
return number
as
c number;
BEGIN
c:= a+b;
return c;
END;
/
```

o/p Function created

```
DECLARE
a number:= &a;
b number:= &b;
c number;
BEGIN
select fdemo(a,b) into c from dual;
dbms_output.put_line('ADDITION='||c);
END;
/
```

o/p Enter value for a: 10
old 2: a number:= &a;
new 2: a number:= 10;
Enter value for b: 20
old 3: b number:= &b
new 3: b number:= 20;
ADDITION = 30

PL/SQL procedure successfully completed

3 Create a function to find square of a number

```
CREATE OR REPLACE FUNCTION fdemo(n number)
return number
as
a number;
BEGIN
a:=n*n;
return a;
END;
/
```

o/p Function created

```
DECLARE
a number:=&a;
b number;
BEGIN
b:=fdemo(a);
dbms_output.put_line('square='||b);
END;
/
```

o/p Enter value for a: 5
old 2: a number:=&a;
new 2: a number:=5;
square=25

PL/SQL procedure successfully completed

Amruta Shinde
219445

4 Create a function to find factorial of a number given by the user

```
CREATE OR REPLACE FUNCTION fdemo(n number)
return number
as
fact number:=1;
BEGIN
for i in 1..n
loop
fact:=fact*i;
end loop;
return fact;
END;
/
```

o/p Function created

```
DECLARE
n number:=&n;
f number;
BEGIN
f:=fdemo(n);
dbms_output.put_line('factorial='||f);
END;
/
```

o/p Enter value for n:5
old 2: n number:=&n;
new 2: n number:=5;
factorial=120

PL/SQL procedure successfully completed

5 Create a function to display the number of employee's from EMP table

```
CREATE OR REPLACE FUNCTION fdemo
return number
as
n number;
BEGIN
select count(*) into n from emp;
return n;
END;
/
```

o/p Function created

```
declare
n number;
begin
n:=fdemo;
dbms_output.put_line('Total Employee='||n);
END;
/
```

o/p Total Employee=14

PL/SQL procedure successfully completed

6. Create a function to Display a message 'Hello World' & return the value with help of a variable

```
CREATE OR REPLACE function fdemo
return varchar
as
BEGIN
```

```
return 'HELLO WORLD';
END;
/
```

o/p function created

```
Select fdemo from dual;
```

o/p FDEMO
HELLO WORLD

c] Creating Packages:-

1 Create a package pkg_circle to calculate the area & circumference of a circle when radius is given by the user

```
CREATE OR REPLACE PACKAGE pkg_circle
AS
 FUNCTION getCircumference(p_radius IN NUMBER)
 RETURN NUMBER;

 FUNCTION getArea(p_radius IN NUMBER)
 RETURN NUMBER;
END;
/
```

o/p Package created

```
CREATE OR REPLACE PACKAGE BODY pkg_circle
AS
g_pi NUMBER := 3.14159
FUNCTION getCircumference(p_radius IN NUMBER)
RETURN NUMBER
IS
BEGIN
```

```
    RETURN 2*g_pi*p_radius;
END;
FUNCTION getArea(p_radius IN NUMBER)
 RETURN NUMBER
IS
 BEGIN
 RETURN g_pi*p_radius*p_raddius;
 END;
END;
/
```

O/P Package body created

```
DECLARE
 r int:=&r;
BEGIN
 dbms_output.put_line(pkg_circle.getCircumference(r));
 dbms_output.put_line(pkg_circle.getArea(r));
END;
/
```

O/P Enter value for r: 2
old 2: r int:=&r;
new 2: r int:=2;
12.56636
12.56636
PL/SQL procedure successfully completed

Amruta Shinde
21945

PAGE NO.
DATE / /

(20)

Sr no	Date	Topic:
19	5-10-21	PL/SQL Control structure

PL/SQL control structure

1 write PL/SQL program to accept an years from user & check whether if is leap year or not

```
DECLARE
    v-year number(4);
  BEGIN
    v-year:=& v-year;
    If mod (v-year, 4) = 0 then
       dbms_output.put_line('leap year');
    Else
       dbms_output.put_line('not a leap year');
    END If;
END;
/
```

O/P
```
Enter value for v_year: 2012
leap year
```

2 write a PL/SQL program to accept no from user & display greatest of two number

```
DECLARE
    V_a number(2);
    V_b number(2);
BEGIN
    V_a:=& v_a;
```

```
v_b := &v_b;
If (v_a > v_b) then
    dbms_output.put_line ('First number is greatest');
Else if (v_b > v_a) then
    dbms_output.put_line ('second number is greatest');
ELSE
    dbms_output.put_line ('Both are equal');
ENDIf;
END;
/
```

```
o/P Enter value for v_a : 5
Enter value for v_b : 6
second number is greatest
```

3. write a PL/SQL block to check whether a number is prime or not

```
DECLARE
   v_number number(0);
   v_flag number(1) := 0;
   v_count number(2) := 2
   BEGIN
    v_number := &v_number;
    v_flag while v_count >= v_number loop
    If mod (v_number, v_count) = 0 Then
       v_flag := 1;
     Exit;
   END If;
   number := number - 1;
   END LOOP;
```

```
If (flag=0) Then
    dbms_output.put_line('x');
END If;
x:=x+1;
END LOOP
END;
/
```

O/P 1, 3, 5, 7, ... 97

Write a program in PLSQL to accept number from user & print its factorial

```
DECLARE
    V_number number(2);
    V_fact number(5):=1;
  BEGIN
    V_number:=&V_number;
    LOOP
    V_fact:=V_fact*V_number;
    V_number:=V_number-1
    Exit when V_number=0;
    END LOOP;
    dbms_output.put_line('Factorial is :'|| fact);
    END;
    /
```

O/P Enter value for V_number 4
old 2: V_number:=4 V_number;
new 2: V_number:=4;
factorial is 24

Amruta Vilas Shinde
219445

1.Create a function that compare two given number & display which one is greater than the other

```
CREATE OR REPLACE FUNCTION GREATES( n1 IN NUMBER, n2 IN NUMBER)
RETURN NUMBER
IS
BEGIN
IF n1<n2 THEN RETURN n2;
ELSE IF n2<n1 THEN RETURN n1;
ELSE
dbms_output.put_line('Both numbers are equal');
END IF  ;
END IF;
END;
/
```

```
SQL> CREATE OR REPLACE FUNCTION GREATES( n1 IN NUMBER, n2 IN NUMBER)
  2  RETURN NUMBER
  3  IS
  4  BEGIN
  5  IF n1<n2 THEN RETURN n2;
  6  ELSE IF n2<n1 THEN RETURN n1;
  7  ELSE
  8  dbms_output.put_line('Both numbers are equal');
  9  END IF  ;
 10  END IF;
 11  END;
 12  /

Function created.
```

Amruta Vilas Shinde
219445

```
DECLARE
n1 NUMBER;
n2 NUMBER;
o NUMBER;
BEGIN
n1:=&n1;
n2:=&n2;
o:=GREATEST(n1,n2);
dbms_output.put_line('GREATER NUMBER IS:'||o);
END;
/
```

```
SQL> DECLARE
  2  n1 NUMBER;
  3  n2 NUMBER;
  4  o NUMBER;
  5  BEGIN
  6  n1:=&n1;
  7  n2:=&n2;
  8  o:=GREATEST(n1,n2);
  9  dbms_output.put_line('GREATER NUMBER IS:'||o);
 10  END;
 11  /
Enter value for n1: 46
old   6: n1:=&n1;
new   6: n1:=46;
Enter value for n2: 89
old   7: n2:=&n2;
new   7: n2:=89;
GREATER NUMBER IS:89

PL/SQL procedure successfully completed.
```

Amruta Vilas Shinde
219445

2.Create a function that calculates factorial of a given positive number

```
CREATE OR REPLACE FUNCTION fn(n IN NUMBER)
RETURN NUMBER
IS
B NUMBER;
BEGIN
B:=1;
For I in 1..n
LOOP
B:=B*I;
END LOOP;
RETURN B;
END;
/
```

```
SQL> CREATE OR REPLACE FUNCTION fn(n IN NUMBER)
  2  RETURN NUMBER
  3  IS
  4  B NUMBER;
  5  BEGIN
  6  B:=1;
  7  For I in 1..n
  8  LOOP
  9  B:=B*I;
 10  END LOOP;
 11  RETURN B;
 12  END;
 13  /

Function created.
```

Amruta Vilas Shinde
219445

```
DECLARE
n NUMBER:=&n;
y NUMBER;
BEGIN
y:=fn(n);
dbms_output.put_line('factorial is:'||y);
END;
/
```

```
SQL> DECLARE
  2  n NUMBER:=&n;
  3  y NUMBER;
  4  BEGIN
  5  y:=fn(n);
  6  dbms_output.put_line('factorial is:'||y);
  7  END;
  8  /
Enter value for n: 4
old   2: n NUMBER:=&n;
new   2: n NUMBER:=4;
factorial is:24

PL/SQL procedure successfully completed.
```

Amruta Vilas Shinde
219445

3.Create procedure that accepts a value from the user & check whether it is prime or not.

```
CREATE OR REPLACE PROCEDURE iprime(p IN NUMBER)
IS
a NUMBER:=1;
b NUMBER;
BEGIN
b:=0;
for i in 2..p-1 loop
IF mod(p,i)=0 THEN
dbms_output.put_line('prime number');
ELSE
dbms_output.put_line('not a prime number');
END IF;
END LOOP;
END;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE iprime(p IN NUMBER)
  2  IS
  3  a NUMBER:=1;
  4  b NUMBER;
  5  BEGIN
  6  b:=0;
  7  for i in 2..p-1 loop
  8  IF mod(p,i)=0 THEN
  9  dbms_output.put_line('prime number');
 10  ELSE
 11  dbms_output.put_line('not a prime number');
 12  END IF;
 13  END LOOP;
 14  END;
 15  /

Procedure created.
```

Amruta Vilas Shinde
219445

```
DECLARE
cn NUMBER:=&cn;
BEGIN
iprime(cn);
END;
/
```

```
SQL> DECLARE
  2  cn NUMBER:=&cn;
  3  BEGIN
  4  iprime(cn);
  5  END;
  6  /
Enter value for cn: 4
old   2: cn NUMBER:=&cn;
new   2: cn NUMBER:=4;
prime number
not a prime number

PL/SQL procedure successfully completed.

SQL> DECLARE
  2  cn NUMBER:=&cn;
  3  BEGIN
  4  iprime(cn);
  5  END;
  6  /
Enter value for cn: 3
old   2: cn NUMBER:=&cn;
new   2: cn NUMBER:=3;
not a prime number

PL/SQL procedure successfully completed.
```

Amruta Vilas Shinde
219445

4.Write pl/sql code to calculate area of circle for value for values of radius 2 to 7 & corresponding values for calculated area should be entered in an empty table :- area (radius, cir_area).

CREATE TABLE area(radius NUMBER, cir_area int);

```
SQL> CREATE TABLE area(radius NUMBER, cir_area int);

Table created.

SQL>
```

```
DECLARE
pi NUMBER :=3.14;
r NUMBER;
c_area NUMBER;
BEGIN
For r in 2..7 LOOP
c_area:=pi*r*r;
INSERT INTO area(radius,cir_area)values(r,c_area);
END LOOP;
END;
/
```

Amruta Vilas Shinde

219445

```
SQL> DECLARE
  2  pi NUMBER :=3.14;
  3  r NUMBER;
  4  c_area NUMBER;
  5  BEGIN
  6  For r in 2..7 LOOP
  7  c_area:=pi*r*r;
  8  INSERT INTO area(radius,cir_area)values(r,c_area);
  9  END LOOP;
 10  END;
 11  /

PL/SQL procedure successfully completed.

SQL> select * from area;

    RADIUS   CIR_AREA
---------- ----------
         2         13
         3         28
         4         50
         5         79
         6        113
         7        154

6 rows selected.
```

5.Write a procedure to accept a number from user & print whether it is even or odd.

```
CREATE OR REPLACE PROCEDURE e_o(x IN NUMBER)
IS
x2 NUMBER;
BEGIN
x2:=x
IF mod(x2,2)=0 THEN
DBMS_OUTPUT.PUT_LINE('Given Number is EVEN');
ELSE
DBMS_OUTPUT.PUT_LINE('Given Number is ODD');
END IF;
END;
/
```

Amruta Vilas Shinde

219445

```
SQL> CREATE OR REPLACE PROCEDURE e_o(x IN NUMBER)
  2  IS
  3  x2 NUMBER;
  4  BEGIN
  5  x2:=x;
  6  IF mod(x2,2)=0 THEN
  7  DBMS_OUTPUT.PUT_LINE('Given Number is EVEN');
  8  ELSE
  9  DBMS_OUTPUT.PUT_LINE('Given Number is ODD');
 10  END IF;
 11  END;
 12  /

Procedure created.
```

DECLARE

n NUMBER:=&x;

BEGIN

e_o(n);

END;

/

```
SQL> DECLARE
  2  n NUMBER:=&x;
  3  BEGIN
  4  e_o(n);
  5  END;
  6  /
Enter value for x: 3
old   2: n NUMBER:=&x;
new   2: n NUMBER:=3;
Given Number is ODD

PL/SQL procedure successfully completed.

SQL> DECLARE
  2  n NUMBER:=&x;
  3  BEGIN
  4  e_o(n);
  5  END;
  6  /
Enter value for x: 2
old   2: n NUMBER:=&x;
new   2: n NUMBER:=2;
Given Number is EVEN

PL/SQL procedure successfully completed.
```

Amruta Vilas Shinde
219445

6.Create procedure to calculate Fibonacci series.

```
CREATE OR REPLACE PROCEDURE fibo(n IN NUMBER)
IS
a NUMBER(2):=0;
b NUMBER(2):=1;
c NUMBER(2);
BEGIN
IF n<2 THEN
DBMS_OUTPUT.PUT_LINE('Entered Number Should be Greater than 1');
END IF;
DBMS_OUTPUT.PUT_LINE('The Fibinocci Series Is:');
DBMS_OUTPUT.PUT_LINE(a);
DBMS_OUTPUT.PUT_LINE(b);
FOR i in 3..n
LOOP
c:=a+b;
a:=b;
b:=c;
DBMS_OUTPUT.PUT_LINE(c);
END LOOP;
END;
/
```

Amruta Vilas Shinde

219445

```
SQL> CREATE OR REPLACE PROCEDURE fibo(n IN NUMBER)
  2  IS
  3  a NUMBER(2):=0;
  4  b NUMBER(2):=1;
  5  c NUMBER(2);
  6  BEGIN
  7  IF n<2 THEN
  8  DBMS_OUTPUT.PUT_LINE('Entered Number Should be Greater than 1');
  9  END IF;
 10  DBMS_OUTPUT.PUT_LINE('The Fibinocci Series Is:');
 11  DBMS_OUTPUT.PUT_LINE(a);
 12  DBMS_OUTPUT.PUT_LINE(b);
 13  FOR i in 3..n
 14  LOOP
 15  c:=a+b;
 16  a:=b;
 17  b:=c;
 18  DBMS_OUTPUT.PUT_LINE(c);
 19  END LOOP;
 20  END;
 21  /

Procedure created.
```

DECLARE

n NUMBER(2):=&n;

BEGIN

fibo(n);

END;

/

Amruta Vilas Shinde
219445

```
Procedure created.

SQL> DECLARE
  2  n NUMBER(2):=&n;
  3  BEGIN
  4  fibo(n);
  5  END;
  6  /
Enter value for n: 4
old   2: n NUMBER(2):=&n;
new   2: n NUMBER(2):=4;
The Fibinocci Series Is:
0
1
1
2

PL/SQL procedure successfully completed.

SQL> DECLARE
  2  n NUMBER(2):=&n;
  3  BEGIN
  4  fibo(n);
  5  END;
  6  /
Enter value for n: 10
old   2: n NUMBER(2):=&n;
new   2: n NUMBER(2):=10;
The Fibinocci Series Is:
0
1
1
2
3
5
8
13
21
34

PL/SQL procedure successfully completed.
```