

## DATABASE SYSTEMS

### Lab Digital Assignment-3

Name: Mirdhula

Course Code: BCSE302P

#### Section A (2 Marks)

Aim: To understand the concept of views and create virtual tables. To perform necessary operations on the virtual tables.

- a. Create a View for the employee table with selected attributes (any 3 attributes)

```
SQL> create view bce3466db_view AS
  2 select emp_id,f_name,salary,deptno from bce3466db;

View created.

SQL> select *from bce3466db_view;
```

EMP_ID	F_NAME	SALARY	DEPTNO
2031	Soumya	12000	101
3420	Aditya	15000	103
2793	Praneesh	14000	102
3201	Claire	14000	103
2954	Manish	13000	101
3003	Amina	21000	104
2980	Samiya	12500	101
3478	Shaurya	20000	104
3120	Hema	14000	102

9 rows selected.

- b. Create a view for student database for all those students who have register for the course BCSE303P

```
SQL> create view studfull_view AS select*from student where course='BCSE303P';

View created.

SQL> select*from studfull_view;
```

ID	NAME	COURSE	MARKS
21	Sarah Mathew	BCSE303P	67
23	Kamal Zaid	BCSE303P	82
24	Naina Palkar	BCSE303P	77
20	Ashish Dua	BCSE303P	89

c. Create a view for the student database by having the course code and the average marks secured by the students for each of the subjects registered

```
SQL> create view student_view AS select course, avg(marks) AS avg
  2  From student group by course;

View created.

SQL> select *from student_view
  2  ;
```

COURSE	AVG
BCSE302P	62
BCSE308P	59
BCSE303P	78.75

d. Is it possible to update the content of source database with the use of view definition made through the source database? Justify your answer with an example illustration.

Yes, using a view definition cannot directly alter the data in a source database.

A view is a virtual table that is formed as a result of a query that obtains data from one or more underlying table.

```
SQL> insert into studfull_view values(26, 'Mary James', 'BCSE303P', 69);

1 row created.

SQL> select*from studfull_view;
```

ID	NAME	COURSE	MARKS
21	Sarah Mathew	BCSE303P	67
23	Kamal Zaid	BCSE303P	82
24	Naina Palkar	BCSE303P	77
20	Ashish Dua	BCSE303P	89
26	Mary James	BCSE303P	69

### Section B (2 Marks)

Aim: To create necessary subqueries to perform the required operations in the employee table.

- a. Find the names of the employees whose salary is greater than the average salary of all the employees in every department

```
SQL> select e.name from bce3466db e where e.salary > (select avg(salary)
2 From bce3466db where deptno=e.deptno);

NAME
-----
Aditya Tyagi
Manish Malhotra
Amina Faizan
Diana Dsouza
Harish Gyan
```

- b. Give an example to illustrate with a query that returns an error message as: ORA-01427:  
a single-row subquery returns more than one row

```
SQL> select E.name
2 from employee_bce0544 E
3 where E.salary > (
4     select AVG(E2.salary)
5     FROM employee_bce0544 E2
6     INNER JOIN department_0544 D ON E2.emp_id=D.dept_id
7     GROUP BY D.dept_id
8 );
      select AVG(E2.salary)
      *
ERROR at line 4:
ORA-01427: single-row subquery returns more than one row
```

- c. Make use of the membership operators to perform the following query: Print all the employee id and employee name for those who are not working in sales or production department.

```
SQL> select e.emp_id,e.name from bce3466db e
2 left join dept d on e.deptno=d.dept_no
3 where d.dept_name NOT IN('Design','Marketing');

EMP_ID NAME
-----
2031 Soumya Mishra
2954 Manish Malhotra
3003 Amina Faizan
2980 Samiya Murtaza
3478 Sundar Pichai
3340 Diana Dsouza
3216 Sweta Mohan
2432 Harish Gyan

3 rows selected.
```

d. Find all those employee names who are working in the project “Coderview” and works for more than 10 hours per week. (you must use employee, assignment and project table)

SQL> select e.name from bce3466db e  
2 inner join emp\_proj EP on e.emp\_id=ep.empid  
3 inner join proj3466 P on ep.projid=p.proj\_id  
4 inner join dept D on e.deptno=d.dept\_no  
5 where p.proj\_name='Coderview' AND EP.hrs>10;  
  
NAME  
-----  
Soumya Mishra  
Aditya Tyagi  
Sundar Pichai  
Diana Dsouza  
  
SQL> select\*from proj3466;

EMPLOYEE\_PROJECT TABLE

EMPID	PROJID	HRS
2031	2	15
3420	2	20
2793	1	25
3201	3	20
2954	1	30
3003	3	20
2980	1	25
3478	2	20
3120	3	30
3340	2	15
3216	1	10

EMPID	PROJID	HRS
2432	3	15

PROJ_ID	PROJ_NAME	LOC
1	SCOPE	Mumbai
2	Coderview	New Jersey
3	SELECT	Delhi

12 rows selected.

### Section C (6 Marks)

**Aim: To create PL/SQL block for the given questions with respect to the schema given below**

<pre>SQL&gt; desc dept2;</pre>		
Name	Null?	Type
DEPT_NAME		VARCHAR2(15)
DEPTNO		NUMBER(5)
MGR_SSN		CHAR(9)
MGR_START		DATE
<pre>SQL&gt; desc emp3466;</pre>		
Name	Null?	Type
FNAME		VARCHAR2(15)
MIDNAME		CHAR(2)
LNAME		VARCHAR2(15)
SSN		CHAR(9)
BDAY		DATE
ADDRESS		VARCHAR2(50)
SEX		CHAR(1)
SALARY		NUMBER(7)
SUPER_SSN		CHAR(9)
DEPTNO		NUMBER(5)

- a. Write a PL/SQL block to print even number ranging between 1 and 100 in reverse order.

```
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2  counter NUMBER := 100;
  3  BEGIN
  4  WHILE counter >= 1 LOOP
  5  IF MOD(counter, 2) = 0 THEN
  6  DBMS_OUTPUT.PUT_LINE(counter);
  7  END IF;
  8  counter := counter - 1;
  9  END LOOP;
 10  END;
 11  /
```

100	48
98	46
96	44
94	42
92	40
90	38
88	36
86	34
84	32
82	30
80	28
78	26
76	24
74	22
72	20
70	18
68	16
66	14
64	12
62	10
60	8
58	6
56	4
54	2
52	
50	

b. Write a PL/SQL block to change address of a particular employee by taking his/her employee number interactively.

```
SQL> DECLARE
  2  v_emp_num NUMBER;
  3  v_new_address VARCHAR2(50);
  4  BEGIN
  5  DBMS_OUTPUT.PUT('Enter Employee Number: ');
  6  v_employee_number := &employee_number;
  7  DBMS_OUTPUT.PUT('Enter New Address: ');
  8  v_new_address := '&new_address';
  9  UPDATE employee_bce0544
 10 SET Address = v_new_address
 11 WHERE Employee_Number = v_employee_number;
 12 DBMS_OUTPUT.PUT_LINE('Address updated successfully for Employee Number
 13 v_employee_number);
 14 EXCEPTION
 15 WHEN OTHERS THEN
 16 DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
 17 END;
 18 /
Enter value for employee_number: 4
old 6: v_employee_number := &employee_number;
new 6: v_employee_number := 4;
Enter value for new_address: 19 Pine st;
old 8: v_new_address := '&new_address';
new 8: v_new_address := '19 Pine st';
```

c. Write a PL/SQL block to display number of employees for each department.

```
SQL> DECLARE
  2  v_department_number NUMBER;
  3  v_employee_count NUMBER;
  4  BEGIN
  5  FOR dept_rec IN (SELECT DISTINCT dept_no FROM
  6 employee2_bce0544) LOOP
  7  v_department_number := dept_rec.Dept_no;
  8  SELECT COUNT(*) INTO v_employee_count
  9  FROM employee2_bce0544
 10 WHERE dept_no = v_department_number;
 11 -- Displaying the department number and the corresponding employee count
 12 DBMS_OUTPUT.PUT_LINE('Department Number: ' || v_department_number || ',
 13 Employee Count: ' || v_employee_count);
 14 END LOOP;
 15 EXCEPTION
 16 WHEN OTHERS THEN
 17 DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
 18 END;
 19 /
Department Number: 1003,
Employee Count: 2
Department Number: 1001,
Employee Count: 2
Department Number: 1002,
Employee Count: 3
PL/SQL procedure successfully completed.
```



- d. Write a program to delete employee details who are having age >60.

```
SQL> DECLARE
  2  v_today DATE := SYSDATE;
  3  v_age_limit NUMBER := 60;
  4  BEGIN
  5  DELETE FROM employee2_bce0544
  6  WHERE TRUNC(MONTHS_BETWEEN(v_today, dob)/ 12) > v_age_limit;
  7  DBMS_OUTPUT.PUT_LINE('Employee details of those aged over 60 have been
  8  deleted.');
```

Employee details of those aged over 60 have been deleted.

- e. Write a PL/SQL block to display employees who are top three earners in the company.

```
SQL> DECLARE
  2  v_counter NUMBER := 0;
  3  BEGIN
  4  FOR top_earner_rec
  5  IN (
  6  SELECT fname, lname, salary
  7  FROM (
  8  SELECT fname, lname, salary, RANK() OVER (ORDER BY
  9  Salary DESC) AS r
  10 FROM emp3466
  11 )
  12 WHERE r <= 3
  13 ) LOOP
  14 v_counter := v_counter + 1;
  15 DBMS_OUTPUT.PUT_LINE('Employee' || v_counter || ': ' ||
  16 top_earner_rec.Firstname || ' || top_earner_rec.Lastname ||
  17 ', Salary: ' || top_earner_rec.Salary);
  18 END LOOP;
  19 EXCEPTION
  20 WHEN OTHERS THEN
  21 DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
  22 END;
  23 /
```

```
Employee 1: anshika gupta, Salary: 90000
Employee 2: sachin mani, Salary: 90000
Employee 3: bala madhi, Salary: 86000
```

- f. Write a PL/SQL code to print the employee's cadre (Designation) based on their basic scales

```
SQL> DECLARE
  2  v_employee_scale NUMBER;
  3  v_employee_cadre VARCHAR2(50);
  4  BEGIN
  5  v_employee_scale := 1; -- Replace with the desired employee scale
  6  CASE
  7  WHEN v_employee_scale <= 5000 THEN
  8  v_employee_cadre := 'Project head';
  9  WHEN v_employee_scale <= 10000 THEN
  10 v_employee_cadre := 'Associate';
  11 WHEN v_employee_scale <= 15000 THEN
  12 v_employee_cadre := 'Senior Associate';
  13 ELSE
  14 v_employee_cadre := 'Manager';
  15 END CASE;
  16 DBMS_OUTPUT.PUT_LINE('Employee Cadre: ' || v_employee_cadre);
  17 EXCEPTION
  18 WHEN OTHERS THEN
  19 DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
  20 END;
  21 /
```

Employee Cadre: Project head

PL/SQL procedure successfully completed.