Database Management System

EXPERIMENT 11 CREATING VIEWS

NAME: Madhan RV

ROLL: 241801142

1. Create EMPLOYEE_VU view based on employee numbers, names, and department numbers

```
CREATE OR REPLACE VIEW employee_vu AS
SELECT employee_id, first_name, last_name AS employee,
department_id
FROM employees;
```

Expected Output:

```
View created.
```

2. Display the contents of the EMPLOYEE_VU view

```
SELECT * FROM employee_vu;
```

Expected Output:

```
EMPLOYEE_ID FIRST_NAME           EMPLOYEE
DEPARTMENT_ID
----------- -------------------- -------------------- ------
```

```
--------
        100 Steven               King
90
        101 Neena                Kochhar
90
        102 Lex                  De Haan
90
        103 Alexander            Hunold
60
        104 Bruce                Ernst
60
        105 David                Austin
60
        106 Valli                Pataballa
60
        107 Diana                Lorentz
60
        108 Nancy                Greenberg
100
        109 Daniel               Faviet
100
        110 John                 Chen
100
        ... (continues for all employees)
```

3. Select view name and text from USER_VIEWS

```
SELECT view_name, text
FROM user_views
WHERE view_name = 'EMPLOYEE_VU';
```

Expected Output:

```
VIEW_NAME      TEXT
-------------  ------------------------------------------------
-----
EMPLOYEE_VU    SELECT employee_id, first_name, last_name AS
employee, department_id FROM employees
```

4. Using EMPLOYEE_VU view, display all employee names and departments

```
SELECT employee, department_id
FROM employee_vu
ORDER BY department_id;
```

Expected Output:

```
EMPLOYEE                DEPARTMENT_ID
------------------- --------------
Whalen                            10
Fay                               20
Hartstein                         20
Marvis                            30
Baida                             30
Colmenares                        30
Himuro                            30
Khoo                              30
Raphaely                          30
Tobias                            30
... (continues grouped by department)
```

5. Create DEPT50 view with restrictions

```
CREATE OR REPLACE VIEW dept50 AS
SELECT employee_id AS empno, last_name AS employee,
```

```
department_id AS deptno
FROM employees
WHERE department_id = 50
WITH CHECK OPTION CONSTRAINT dept50_ck;
```

Expected Output:

```
View created.
```

6. Display structure and contents of DEPT50 view

```
DESC dept50;
```

Expected Output:

```
Name      Null?    Type
--------- -------- ------------
EMPNO              NUMBER(6)
EMPLOYEE           VARCHAR2(25)
DEPTNO             NUMBER(4)
 SELECT * FROM dept50;
```

Expected Output:

```
    EMPNO EMPLOYEE                       DEPTNO
---------- ------------------------- ---------
       120 Weiss                            50
       121 Fripp                            50
       122 Kaufling                         50
       123 Vollman                          50
       124 Mourgos                          50
       125 Nayer                            50
       126 Mikkilineni                      50
```

```
       127 Landry                          50
       128 Markle                          50
       129 Bissot                          50
       130 Atkinson                        50
       ... (all employees from department 50)
```

7. Attempt to reassign Matos to department 80

```
UPDATE dept50
SET deptno = 80
WHERE employee = 'Matos';
```

Expected Output:

```
ERROR at line 1:
ORA-01402: view WITH CHECK OPTION where-clause violation
```

8. Create SALARY_VU view with employee details, department names, salaries, and grades

```
CREATE OR REPLACE VIEW salary_vu AS
SELECT e.last_name AS "Employee",
       d.department_name AS "Department",
       e.salary AS "Salary",
       jg.grade_level AS "Grade"
FROM employees e
JOIN departments d ON e.department_id = d.department_id
JOIN job_grades jg ON e.salary BETWEEN jg.lowest_sal AND
jg.highest_sal;
```

Expected Output:

```
View created.
```

Display SALARY_VU view contents:

```
SELECT * FROM salary_vu ORDER BY "Salary" DESC;
```

Expected Output:

```
Employee             Department                  Salary
G
-------------------- ------------------------ ----------
-
King                 Executive                   24000
F
Kochhar              Executive                   17000
E
De Haan              Executive                   17000
E
Hartstein            Marketing                   13000
D
Russell              Sales                       14000
D
Partners             Sales                       13500
D
Errazuriz            Sales                       12000
D
Higgins              Accounting                  12000
D
... (continues with all employees)
```

Additional View Examples:

Simple View with Column Aliases

```
CREATE OR REPLACE VIEW emp_simple AS
SELECT employee_id AS id,
       first_name || ' ' || last_name AS full_name,
       salary,
       department_id
FROM employees;
```

Complex View with Aggregation

```
CREATE OR REPLACE VIEW dept_summary AS
SELECT d.department_name,
       COUNT(e.employee_id) AS employee_count,
       AVG(e.salary) AS avg_salary,
       MAX(e.salary) AS max_salary,
       MIN(e.salary) AS min_salary
FROM departments d
LEFT JOIN employees e ON d.department_id = e.department_id
GROUP BY d.department_name;
```

Read-Only View

```
CREATE OR REPLACE VIEW emp_readonly AS
SELECT employee_id, first_name, last_name, hire_date
FROM employees
WITH READ ONLY;
```

Key View Concepts Demonstrated:

1. **Simple Views** - Based on single table with no functions
2. **Complex Views** - Contain joins, groups, or functions
3. **Column Aliases** - Renaming columns in views
4. **WITH CHECK OPTION** - Prevents DML that would exclude rows from view

5. **WITH READ ONLY** - Prevents all DML operations on view
6. **Data Dictionary** - USER_VIEWS contains view definitions
7. **View Updating** - Some views allow DML, others don't