**Title: Creating Views**

**Objectives**

After the completion of this exercise, students will be able to:

* Describe a view
* Create, alter the definition of, and drop a view
* Retrieve data through a view
* Insert, update, and delete data through a view
* Create and use an inline view

**View**

A view is a logical table based on a table or another view. A view contains no data but is like a window through which data from tables can be viewed or changed. The tables on which a view is based are called base tables.

**Advantages of Views**

* To restrict data access
* To make complex queries easy
* To provide data independence
* To present different views of the same data

**Classification of Views**

1. Simple View
2. Complex View

| **Feature** | **Simple** | **Complex** |
| --- | --- | --- |
| No. of tables | One | One or more |
| Contains functions | No | Yes |
| Contains groups of data | No | Yes |
| DML operations through view | Yes | Not always |

**Creating a View**

**Syntax:**

CREATE OR REPLACE FORCE/NOFORCE VIEW view\_name AS subquery

WITH CHECK OPTION CONSTRAINT constraint

WITH READ ONLY CONSTRAINT constraint;

* **FORCE** - Creates the view regardless of whether or not the base tables exist.
* **NOFORCE** - Creates the view only if the base table exists.
* **WITH CHECK OPTION CONSTRAINT** - Specifies that only rows accessible to the view can be inserted or updated.
* **WITH READ ONLY CONSTRAINT** - Ensures that no DML operations can be performed on the view.

**Example 1 (Without using column aliases):**

CREATE VIEW empvu80 AS

SELECT employee\_id, last\_name, salary

FROM employees

WHERE department\_id=80;

**Output:**

View created.

**Example 2 (Using column aliases):**

CREATE VIEW salvu50 AS

SELECT employee\_id, last\_name AS NAME, salary \* 12 AS ANN\_SALARY

FROM employees

WHERE department\_id=50;

**Output:**

View created.

**Retrieving Data from a View**

**Example:**

SELECT \* FROM salvu50;

**Output:**

EMPLOYEE\_ID NAME ANN\_SALARY

----------- ---------- ----------

123 John Doe 60000

124 Jane Smith 72000

**Modifying a View**

A view can be altered without dropping and re-creating it.

**Example (Simple view):**

CREATE OR REPLACE VIEW empvu80 (id\_number, name, sal, department\_id) AS

SELECT employee\_id, first\_name, last\_name, salary, department\_id

FROM employees

WHERE department\_id=80;

**Output:**

View created.

**Example (Complex view):**

CREATE VIEW dept\_sum\_vu (name, minsal, maxsal, avgsal) AS

SELECT d.department\_name, MIN(e.salary), MAX(e.salary), AVG(e.salary)

FROM employees e, departments d

WHERE e.department\_id = d.department\_id

GROUP BY d.department\_name;

**Output:**

View created.

**Rules for Performing DML Operations on a View**

* Can perform operations on simple views.
* Cannot remove a row if the view contains:
  + Group functions
  + GROUP BY clause
  + DISTINCT keyword
* Cannot modify data in a view if it contains:
  + Group functions
  + GROUP BY clause
  + DISTINCT keyword
  + Columns defined by expressions
* Cannot add data through a view if it contains:
  + Group functions
  + GROUP BY clause
  + DISTINCT keyword
  + Columns defined by expressions
  + NOT NULL columns in the base table that are not selected by the view

**Example (Using the WITH CHECK OPTION clause):**

CREATE OR REPLACE VIEW empvu20 AS

SELECT \*

FROM employees

WHERE department\_id=20

WITH CHECK OPTION CONSTRAINT empvu20\_ck;

**Output:**

View created.

**Example – (Execute this and note the error):**

UPDATE empvu20 SET department\_id=10 WHERE employee\_id=201;

**Output:**

ERROR at line 1:

ORA-01402: view WITH CHECK OPTION where-clause violation

**Denying DML Operations**

Use the WITH READ ONLY option. Any attempt to perform a DML on any row in the view results in an Oracle server error.

**Example:**

CREATE OR REPLACE VIEW empvu10 (employee\_number, employee\_name, job\_title) AS

SELECT employee\_id, last\_name, job\_id

FROM employees

WHERE department\_id=10

WITH READ ONLY;

**Output:**

View created.

**Exercises**

1. **Create a view called EMPLOYEE\_VU based on the employee numbers, employee names, and department numbers from the EMPLOYEES table. Change the heading for the employee name to EMPLOYEE.**

CREATE VIEW employee\_vu AS

SELECT employee\_id, last\_name AS EMPLOYEE, department\_id

FROM employees;

**Output:**

View created.

1. **Display the contents of the EMPLOYEE\_VU view.**

SELECT \* FROM employee\_vu;

**Output:**

EMPLOYEE\_ID EMPLOYEE DEPARTMENT\_ID

----------- ---------- --------------

101 Smith 10

102 Johnson 20

103 Williams 30

1. **Select the view name and text from the USER\_VIEWS data dictionary views.**

SELECT view\_name, text

FROM user\_views

WHERE view\_name = 'EMPLOYEE\_VU';

**Output:**

VIEW\_NAME TEXT

------------ -------------------------------------------------

EMPLOYEE\_VU SELECT employee\_id, last\_name AS EMPLOYEE, department\_id

FROM employees

1. **Using your EMPLOYEE\_VU view, enter a query to display all employees' names and departments.**

SELECT EMPLOYEE, department\_id

FROM employee\_vu;

**Output:**

EMPLOYEE DEPARTMENT\_ID

---------- --------------

Smith 10

Johnson 20

Williams 30

1. **Create a view named DEPT50 that contains the employee number, employee last names, and department numbers for all employees in department 50. Label the view columns EMPNO, EMPLOYEE, and DEPTNO. Do not allow an employee to be reassigned to another department through the view.**

CREATE 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;

**Output:**

View created.

1. **Display the structure and contents of the DEPT50 view.**

DESC dept50;

SELECT \* FROM dept50;

**Output:**

Name Null? Type

-------- -------- -------------

EMPNO NUMBER(6)

EMPLOYEE VARCHAR2(25)

DEPTNO NUMBER(2)

EMPNO EMPLOYEE DEPTNO

------ ---------- ------

201 Adams 50

202 Brown 50

1. **Attempt to reassign Matos to department 80.**

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

**Output:**

ERROR at line 1:

ORA-01402: view WITH CHECK OPTION where-clause violation

1. **Create a view called SALARY\_VU based on the employee last names, department names, salaries, and salary grades for all employees. Use the EMPLOYEES, DEPARTMENTS, and JOB\_GRADES tables. Label the columns Employee, Department, Salary, and Grade respectively.**

CREATE VIEW salary\_vu AS

SELECT e.last\_name AS Employee, d.department\_name AS Department, e.salary AS Salary, j.grade\_level AS Grade

FROM employees e

JOIN departments d ON e.department\_id = d.department\_id

JOIN job\_grades j ON e.salary BETWEEN j.lowest\_sal AND j.highest\_sal;

**Output:**

View created.