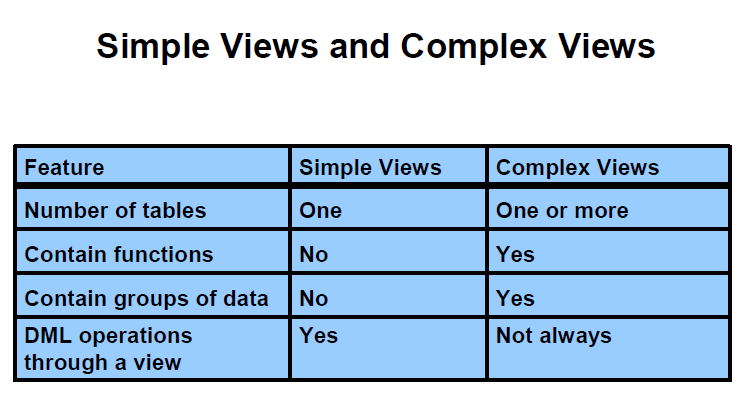
# CHAPTER 4

1. **Create simple & create complex view**



There are two classifications for views: simple and complex. The basic difference is related to the DML (INSERT, UPDATE, and DELETE) operations.

• A simple view is one that:

- Derives data from only one table

- Contains no functions or groups of data

- Can perform DML operations through the view

• A complex view is one that:

- Derives data from many tables

- Contains functions or groups of data

- Does not always allow DML operations through the view

Syntax:

**CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW *view***

**[(*alias*[, *alias*]...)]**

**AS *subquery***

**[WITH CHECK OPTION [CONSTRAINT *constraint*]]**

**[WITH READ ONLY [CONSTRAINT *constraint*]];**

In the syntax:

OR REPLACE re-creates the view if it already exists

FORCE creates the view regardless of whether or not the base tables exist

NOFORCE creates the view only if the base tables exist (This is the default.)

view is the name of the view

alias specifies names for the expressions selected by the view’s query. (The number of aliases must match the number of expressions selected by the view.)

subquery is a complete SELECT statement (You can use aliases for the

columns in the SELECT list.)

WITH CHECK OPTION specifies that only those rows that are accessible to the view can be inserted or updated

constraint is the name assigned to the CHECK OPTION constraint

WITH READ ONLY ensures that no DML operations can be performed on this view

Example:

* Simple view

**CREATE** **VIEW** empvu80

**AS** **SELECT** employee\_id, last\_name, salary

**FROM** employees

**WHERE** department\_id = 80;

**CREATE** **VIEW** salvu50

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

**FROM** employees

**WHERE** department\_id = 50;

**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;

\* WITH CHECK OPTION

**CREATE** **OR** **REPLACE** **VIEW** empvu20

**AS** **SELECT** \*

**FROM** employees

**WHERE** department\_id = 20

**WITH** **CHECK** **OPTION** **CONSTRAINT** empvu20\_ck ;

If you run:

**UPDATE** empvu20

SET department\_id = 10

**WHERE** employee\_id = 201;

ERROR at line 1:

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

\* WITH READ ONLY

**CREATE** **OR** **REPLACE** **VIEW** empvu10

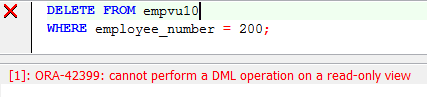
(employee\_number, employee\_name, job\_title)

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

**FROM** employees

**WHERE** department\_id = 10

**WITH** **READ** **ONLY** ;



* Complex view

**CREATE** **OR** **REPLACE** **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 **JOIN** departments d

**ON** (e.department\_id = d.department\_id)

**GROUP** **BY** d.department\_name;

Note:

\* You can perform DML operations on data through a view if those operations follow certain rules.

You can remove a row from a view unless it contains any of the following:

• Group functions

• A GROUP BY clause

• The DISTINCT keyword

• The pseudocolumn ROWNUM keyword

\* You cannot modify data in a view if it contains:

• Group functions

• A GROUP BY clause

• The DISTINCT keyword

• The pseudocolumn ROWNUM keyword

\* You cannot add data through a view if the view includes:

• Group functions

• A GROUP BY clause

• The DISTINCT keyword

• The pseudocolumn ROWNUM keyword

• Columns defined by expressions

• NOT NULL columns in the base tables that are not selected by the view

1. **Drop view**

Syntax:

**DROP VIEW *view*;**

Example:

**DROP** **VIEW** empvu80;

1. **Using view**

You can retrieve data from a view as you would from any table.

Syntax:

**SELECT \*|{[DISTINCT] *column*|*expression* [*alias*],...}**

**FROM *table;***

Example:

Select \* from empvu80

1. **Practices**
   1. The staff in the HR department wants to hide some of the data in the EMPLOYEES table. They want a view called vw\_EMPLOYEES based on the employee numbers, employee names, and department numbers from the EMPLOYEES table. They want the heading for the employee name to be EMPLOYEE.
   2. Confirm that the view works. Display the contents of the vw\_EMPLOYEES view.
   3. Using your vw\_EMPLOYEES view, write a query for the HR department to display all employee names and department numbers.
   4. Department 50 needs access to its employee data. Create a view named DEPT50 that contains the employee numbers, employee last names, and department numbers for all employees in department 50. You have been asked to label the view columns EMPNO, EMPLOYEE, and DEPTNO. For security purposes, do not allow an employee to be reassigned to another department through the view.
   5. Test your view. Attempt to reassign Matos to department 80.