USMAN INSTITUTE OF TECHNOLOGY

Department of Computer Science

CS311 Introduction to Database Systems

Lab#9

Objective:

- **Creating Views**

Name of Student: Muhammad Waleed

Roll No: 20B-115-SE **Sec.** B

Date of Experiment: _____7-01-2022_____

.....

Marks Obtained/Remarks: _____

Signature: _____

Creating Views

What is a View?

A view is a logical table based on other tables or another view. A view contains no data of its own but is like a window through which data from tables can be viewed or changed. The tables in which a view is based are called *base tables*. The view is stored as a SELECT statement in the data dictionary.

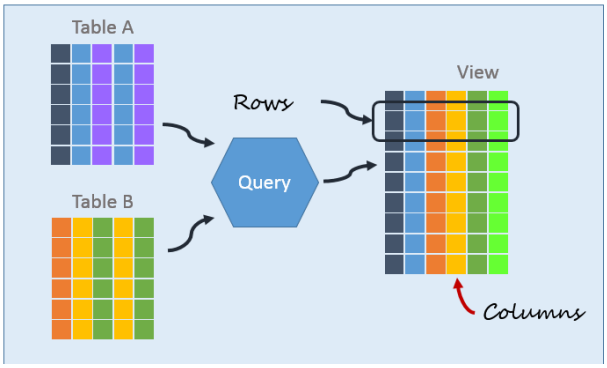


Figure 9.1

Why Use Views?

- Views can represent a subset of the data contained in a table. Consequently, a view can limit the degree of exposure of the underlying tables to the outer world: a given user may have permission to query the view, while denied access to the rest of the base table.
- Views can join and simplify multiple tables into a single virtual table.
- Views can act as aggregated tables, where the database engine aggregates data (sum, average, etc.) and presents the calculated results as part of the data.
- Views can hide the complexity of data. For example, a view could appear as Sales2000 or Sales2001, transparently partitioning the actual underlying table.
- Views take very little space to store; the database contains only the definition of a view, not a copy of all the data that it presents.
- Depending on the SQL engine used, views can provide extra security.
- To present different views of the same data to different users.

Simple and Complex Views

There are two classifications for views:

1. Simple
2. Complex.

The basic difference is related to the DML (insert, update and delete) operations.

Table 9.1

Feature	Simple Views	Complex Views
Number of tables	One	One or more
Contain functions	No	Yes
Contain groups of data	No	Yes
DML through view	Yes	Not always

Lab No. 9

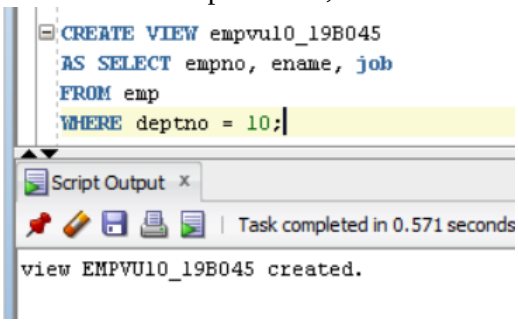
Creating a View

We can create a view by embedding a subquery within the CREATE VIEW statement. The syntax is as follows:-

```
CREATE [OR REPLACE] VIEW view
AS subquery;
```

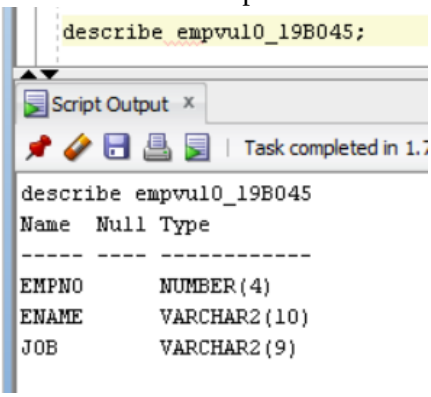
For example, to create a view, EMPVU10, that contains the employee number, name and job title for all the employees in department 10.

```
CREATE VIEW empvu10
AS SELECT empno, ename, job
FROM emp
WHERE deptno = 10;
```



We can display the structure of the view by using the SQL*Plus DESCRIBE command as follows: -

```
DESCRIBE empvu10
```



We can also create views by using column aliases in the subquery.

```
CREATE VIEW salvu30
AS SELECT empno EMPLOYEE_NUMBER, ename NAME, sal SALARY
FROM emp
WHERE deptno = 30;
```

```
CREATE VIEW salvu30_19B045
AS SELECT empno EMPLOYEE_NUMBER, ename NAME, sal SALARY
FROM emp
Where Deptno = 30;
```

Script Output | Task completed in 0.232 seconds

```
view SALVU30_19B045 created.
```

Now select the columns from this view by the given alias names.
The data from the view would be retrieved as follows: -

SELECT *
FROM salvu30;

```
SELECT *
From salvu30_19B045;
```

Query Result | SQL | All Rows Fetched: 6 in 0.165 seconds

	EMPLOYEE_NUMBER	NAME	SALARY
1	7499	ALLEN	1600
2	7521	WARD	1250
3	7654	MARTIN	1250
4	7698	BLAKE	2850
5	7844	TURNER	1500
6	7900	JAMES	950

Views in the data dictionary

Once a view has been created, we can query the data dictionary table called USER_VIEWS to see the name of the view and the view definition. The text of the SELECT statement that constitutes the view is stored in a LONG column.

Creating a Complex View

A complex view contains columns from multiple tables and may also include group functions.

- To create a complex view to show employee number, employee name and department name, we would have to join EMP and DEPT tables as follows:-

```
CREATE VIEW EMP_DEPT
AS
SELECT EMPNO, ENAME, DNAME
FROM EMP, DEPT
WHERE EMP.DEPTNO = DEPT.DEPTNO;
```

```
CREATE VIEW EMP_DEPT_19B045
AS
SELECT EMPNO, ENAME, DNAME
FROM EMP, DEPT
Where Emp.Deptno = Dept.Deptno;
```

Script Output | Task completed in 0.031 seconds

```
view EMP_DEPT_19B045 created.
```

- To create a complex view that contains group functions to display values from two tables.

```
CREATE VIEW dept_sum_vu (name, minsal, maxsal, avgsal)
AS SELECT d.dname, MIN(e.sal), MAX(e.sal), AVG(e.sal)
FROM EMP e, DEPT d
WHERE e.DEPTNO = d.DEPTNO
GROUP BY d.dname;
```

```
CREATE VIEW dept_sum_vu_19B045 (name, minsal, maxsal, avgsal)
AS SELECT d.dname, MIN(e.sal), MAX(e.sal), AVG(e.sal)
FROM EMP e, DEPT d
WHERE e.DEPTNO = d.DEPTNO
Group By D.Dname;
```

Script Output | Task completed in 0.016 seconds

```
view DEPT_SUM_VU_19B045 created.
```

Removing a View

We can remove a view without losing data because a view is based on underlying tables in the database. The syntax is

```
DROP VIEW view;
```

For example to drop the empvu10 view,

```
DROP VIEW empvu10;
```

```
DROP VIEW empvul0_19B045;
```

Script Output | Task completed in 1.

```
view EMPVUl0_19B045 dropped.
```

Updating a View

```
CREATE OR REPLACE VIEW view_name
AS
    SELECT column1,column2, ...
    FROM table_name
    WHERE condition;
```

Example:

Now we want to add the "Category" column to the "Current Product List" view. We will update the view with the following SQL:

```
CREATE OR REPLACE VIEW [Current ProductList]
AS
    SELECT ProductID,ProductName,Category
    FROM Products
    WHERE Discontinued = No;
```

```
CREATE OR REPLACE VIEW ProductList_19B045
As
SELECT ProductID,ProductName,Categoryid
From Products
Where Discontinued = 0;
```

Script Output | Task completed in 0.132 seconds

```
view PRODUCTLIST_19B045 created.
```

Updating a View

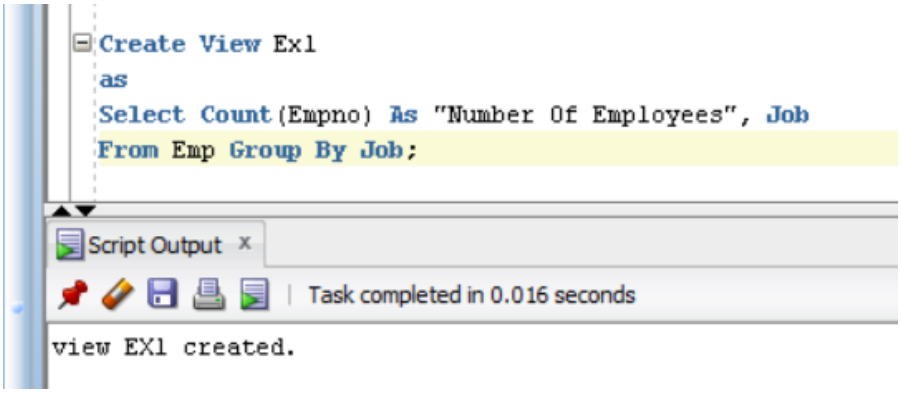
A view can be updated under certain conditions which are given below −

1) The SELECT clause may not contain the keyword DISTINCT.
2) The SELECT clause may not contain summary functions.
3) The SELECT clause may not contain set functions.
4) The SELECT clause may not contain set operators.
5) The SELECT clause may not contain an ORDER BY clause.
6) The FROM clause may not contain multiple tables.
7) The WHERE clause may not contain subqueries.
8) The query may not contain GROUP BY or HAVING.
9) Calculated columns may not be updated.
10) All NOT NULL columns from the base table must be included in the view in order for the INSERT query to function.

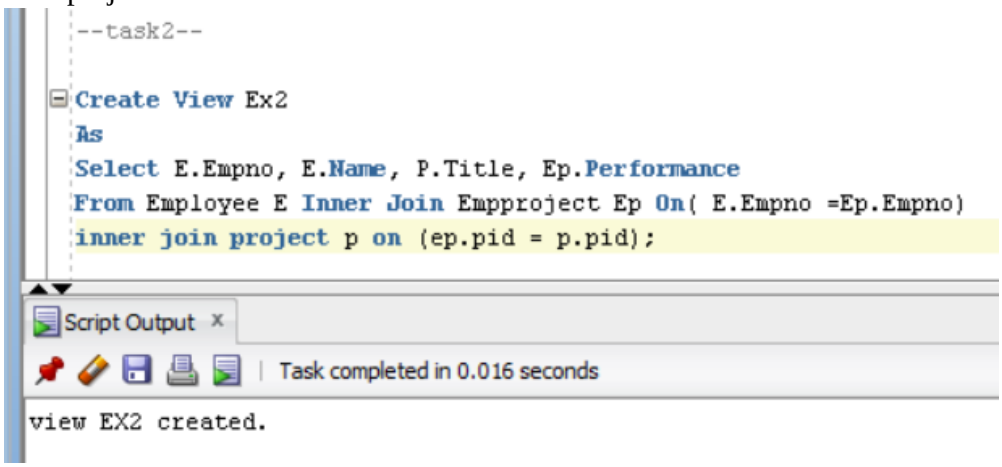
Exercise:

1. Create views for following purposes: -

i. To display each designation and number of employees with that particular designation.



ii. To display employee number, employee name, project title and employee performance in that project.



iii. To display employee number, employee name and number of projects in which employee performance is *excellent*.

```
--task 3--

Create View Ex3
As
Select E.Empno, E.Name, Count(Ep.Pid) As "No of Project"
From Employee E Inner Join Empproject Ep On ( E.Empno = Ep.Empno)
Where Ep.Performance = 'Excellent'
group by e.empno, e.name, ep.pid;
```

Script Output

Task completed in 0.131 seconds

```
view EX3 created.
```

iv. What are the different uses of views? Differentiate between simple and complex views.

Views are use for security purpose, solve complex queries easy, faster response, to strict data access, to provide same data independence.

SIMPLE VIEW:

Simple View in SQL is the view created by involving only single table. In Simple View, only one table is in context hence no major associations need to be applied. We cannot use group functions like MAX(), COUNT(), etc.

COMPLEX VIEW:

Complex view is created on using more than one tables. In Complex View, multiple tables are in the context hence general associations need to be applied like: join conditions, group by clause, order by clause. We can use various group functions

* * * * *