Chapter 10
Creating Views

Objectives

- Describe a view
- Create a view
- Retrieve data through a view
- Alter the definition of a view
- Insert, update, and delete data through a view
- Drop a view

What is a View?

- A view is a customized representation of data from one or more tables.

Why Use Views?

- To restrict database access
- To make complex queries easy
- To allow data independence
- To present different views of the same data

Creating a View

- You embed a subquery within the CREATE VIEW statement.

```
CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW view
  [(alias[, alias]...)]
AS subquery[WITH CHECK OPTION [CONSTRAINT constraint]]
[WITH READ ONLY]
```

- The subquery can contain complex SELECT syntax.
- The subquery cannot contain an ORDER BY clause.

- REPLACE – re-receate the view if already exists
- FORCE – create the view regardless of existence of base tables
- NOFORCE – base tables must exist (default)
- WITH CHECK OPTION – only rows accessible to the view can be inserted or updated
- WITH READ ONLY – no DML operation on the view

Creating a View

– Create a view, EMPVU10, that contains details of employees in department 10.

```
SQL> CREATE VIEW  empvu10
  2  AS SELECT   empno, ename, job
  3  FROM                       emp
  4  WHERE                      deptno = 10;
View created.
```

– Describe the structure of the view by using the SQL*Plus DESCRIBE command.

```
SQL> DESCRIBE empvu10
```

– Create a view by using column aliases in the subquery.

```
SQL> CREATE VIEW  salvu30
  2  AS SELECT   empno EMPLOYEE_NUMBER, ename NAME,
  3                       sal SALARY
  4  FROM                 emp
  5  WHERE                deptno = 30;
View created.
```

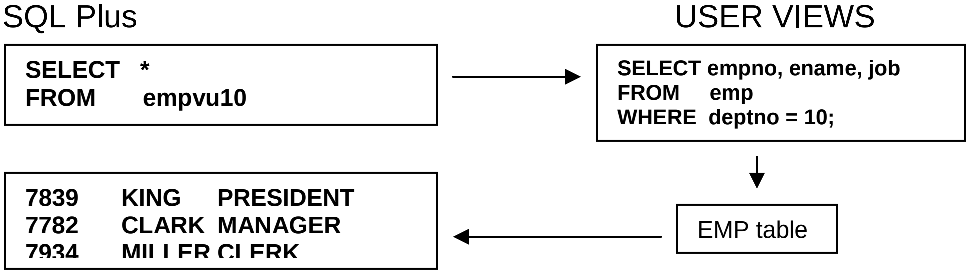
– Select the columns from this view by the given alias names.

Retrieving Data from a View

```
SQL> SELECT *
  2  FROM    salvu30;


EMPLOYEE_NUMBER NAME          SALARY
--------------- ---------- ---------
           7698 BLAKE           2850
           7654 MARTIN          1250
           7499 ALLEN           1600
           7844 TURNER          1500
           7900 JAMES            950
           7521 WARD            1250
6 rows selected.
```

Querying a View



SQL Plus

```
SELECT  *
FROM     empvu10
```

USER VIEWS

```
SELECT empno, ename, job
FROM    emp
WHERE  deptno = 10;
```

EMP table

```
7839     KING     PRESIDENT
7782     CLARK  MANAGER
7934     MILLER CLERK
```

Modifying a View

- Modify the view by using CREATE OR REPLACE VIEW clause. Can add an alias for each column name.

```
SQL> CREATE OR REPLACE VIEW empvu10
  2          (employee_number, employee_name, job_title)
  3  AS SELECT    empno, ename, job
  4  FROM         emp
  5  WHERE        deptno = 10;
View created.
```

- Column aliases in the CREATE VIEW clause are listed in the same order as the columns in the subquery.

Inserting, Updating, and Deleting Data (DML Operations) on a View

- Can update , insert, and delete through a view with restrictions
- If the view is joining more than one table, you can update only one base table at a time.
- For updating and inserting into a view, all columns that are part of a constraint should be in the view definition.
  - USER_UPDATABLE_COLUMNS

Example

```
CREATE OR REPLACE VIEW empvu10
(employee_number, employee_name, salary)
AS SELECT  empno, ename, sal
FROM       emp
WHERE              deptno = 10;

SELECT * FROM empvu10;

SELECT * FROM emp
WHERE deptno=10;

UPDATE empvu10
SET salary = 6000
WHERE employee_name='KING';

SELECT * FROM empvu10;

SELECT * FROM emp
WHERE deptno=10;
```

SELECT table_name, column_name, updatable, insertable, deletable
FROM user_updatable_columns
WHERE table_name = ‘EMPVU10’;

SELECT view_name, text
FROM user_views;

Using Join Views

– Can create a complex view that contains group functions to display values from two tables

```
CREATE VIEW dept_sal_vu(name, avgsal, minsal, maxsal) AS
SELECT d.name, AVG(e.sal), MIN(e.sal),
  MAX(e.sal)
FROM emp e, dept d
WHERE e.deptno = d.deptno
GROUP BY d.dname;
```

Rules for Performing DML operations on a View

- You can perform DML operations on simple views
- You cannot remove a row (or modify or add data in a view) if the view contains the following:
  - Group functions
  - A GROUP BY clause
  - The DISTINCT keyword

- You also cannot modify data in a view if it contains:
  - Columns defined by expressions
  - The ROWNUM pseudocolumn
- You also cannot add data if:
  - There are NOT NULL columns in the base tables that are not selected by the view

Read-Only Views

- WITH READ ONLY specifies the view cannot be updated or deleted and that new rows cannot be inserted
- Very useful for displaying data

Denying DML Operations

– You can ensure that no DML operations occur by adding the WITH READ ONLY option to your view definition.

```
SQL> CREATE OR REPLACE VIEW empvu10
  2        (employee_number, employee_name, job_title)
  3  AS SELECT    empno, ename, job
  4  FROM         emp
  5  WHERE        deptno = 10
  6  WITH READ ONLY;
View created.
```

– Any attempt to perform a DML on any row in the view will result in Oracle Server error.

Using the WITH CHECK OPTION Clause

– Cannot update the columns that join the base tables

– Specifies that inserts and updates should satisfy the WHERE clause of the view

```
CREATE OR REPLACE VIEW TOP_EMP
(emp_no, ename, salary) AS
SELECT emp_no, ename, salary
FROM emp
WHERE salary > 2999
WITH CHECK OPTION CONSTRAINT TOP_EMP_SAL;
```

Cannot add a new employee with a salary less than 3000.

Removing a View

- Remove a view without losing data because a view is based on underlying tables in the database.

```
DROP VIEW view;
```

```
SQL> DROP VIEW empvu10;
View dropped.
```

Inline Views

- Can use a subquery in the FROM clause of a SELECT
- Similar to the way views are used, hence the name “inline views”
- The subquery in the FROM clause is enclosed in parentheses and may be given an alias name.
- Can reference the subquery columns in the parent query

- Give a list of the employee names, their salary, and the average salary in their department. Only want to see those employees whose last name begins with ‘S’.

```
SELECT first_name "First Name",
     last_name "Last Name", salary "Salary",
    TO_CHAR(d1.avg_salary, 'FM$999,999.00') "Average"
FROM employees e1,
      (SELECT department_id,AVG(salary) avg_salary
        FROM employees e2
        GROUP BY department_id) d1
WHERE e1.department_id = d1.department_id
AND upper(last_name) like 'S%';
```

Summary

- A view is derived from data in other tables or other views.
- A view provides the following advantages:
  - Restricts database access
  - Simplifies queries
  - Provides data independence
  - Allows multiple views of the same data
  - Can be dropped without removing the underlying data