Views

This unit is going over Views.

A view is a database objects which will mean that we will have covered two database objects.

We will have covered Table and Views. This will leave the remaining three objects of Sequences, Index and Synonyms yet to cover.

Views

When we finish we should be able to:

1. Describe a view
2. Create a view.
3. Retrieve data through a view.
4. Alter the definition of a view.
5. Insert, update, and delete data through a view.
6. Drop a view.

Purpose of Views

A view is a logical subset or a combination of data from multiple tables. It is based on one or more tables or another view.

A view contains NO DATA of its own but is like a window through which data can be views or changes. The table on which the view is based upon is called the ‘base table’.

Why use Views?

There are several purposed of views:

1. To restrict data access.
2. To make complex queries easy.
3. To allow data independence.
4. To present different view of the same data.
5. It assists us in data security.

Simple and Complex Views?

Simple View.

1. Derives data from only one table.
2. Contains NO functions or Groups of data.
3. Can perform DML through the view, provided integrity constraints are not broken

Complex View.

1. Derives data from many tables.
2. Contains functions or groups of data.
3. Does NOT ALWAYS allow DML through the view.

Creating a Views

CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW view-name [Alias]

AS subquery

[WIITH CHECK OPTION [CONSTRAINT constraint]

[WITH READ ONLY];

These will be explained on the next slide.

Explanation of Views

OR REPLACE:	Re-creates the view if it already exists.
FORCE:	Creates the view regardless of whether or not the base table exists.
NOFORCE:	Created the view ONLY if the base tables exists – This is the default.
view:	The name of the view, usually the name of the table being used ending with VU.

Explanation of Views

alias: Specifies the names for the expressions selected by the view's query.

subquery: Is a complete SELECT statement.

WITH CHECK OPTIONS: Specifies that only rows 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.

Sample View

We want to create a view called empvu10 that contains details of employees in Department 10.

```
CREATE VIEW empvu10
AS SELECT EMPNO, ENAME, JOB
FROM emp
WHERE DEPTNO = 10;
```

Creating a View with Alises

We want to create a view called salvu30 that contains details of employees in Department 30.

```
CREATE VIEW salvu30
AS SELECT EMPNO EMPLOYEE_NUMBER, ENAME NAME, SAL SALARY
FROM emp
WHERE DEPTNO = 30;
```

Retrieving data from a View

In the two previous slides to retrieve the data we would initiate the following select statements.

```
SELECT *
FROM empvu10;
```

and

```
SELECT *
FROM salvu30;
```

Modifying a View

We want to modify empvu10 and add alias for each column name:

```
CREATE OR REPLACE VIEW empvu10
(employee_number, employee_name, job_title)
AS SELECT EMPNO, ENAME, JOB
FROM emp
WHERE DEPTNO = 10;
```

Creating a Complex View

We want to create a complex view that contains group function to display values from two table.

```
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;
```

Rules concerning DML

1. You can perform DML operations on SIMPLE views.
2. You cannot remove or modify a row if the view contains the following:
    a. Group Functions
    b. A GROUP BY clause.
    c. The DISTINCT keyword.
    d. There are NOT NULL columns in the base table that are not selected by the view.

Dropping a view

When a view is dropped we have to realize that NO DATA will be lost since a view CANNOT contain data.

To drop the two previous views we created the command is:

```
DROP VIEW empvu10;

DROP VIEW salvu30;
```