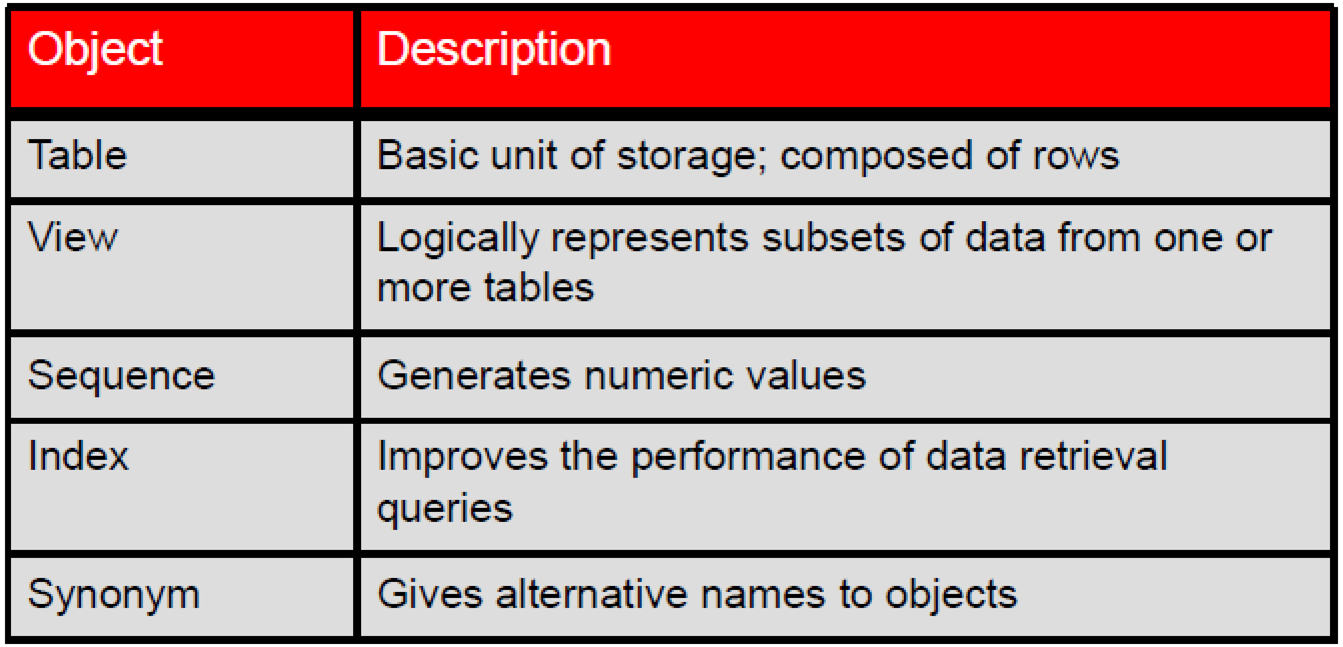
Creating Views

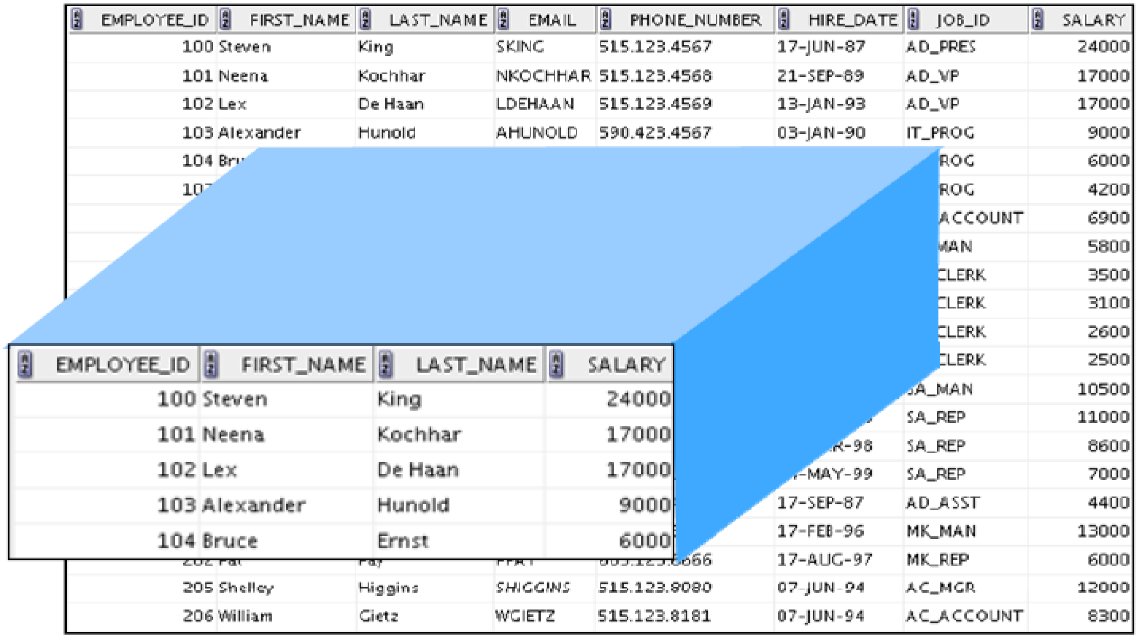
Database Objects



Object	Description
Table	Basic unit of storage; composed of rows
View	Logically represents subsets of data from one or more tables
Sequence	Generates numeric values
Index	Improves the performance of data retrieval queries
Synonym	Gives alternative names to objects

What Is a View?

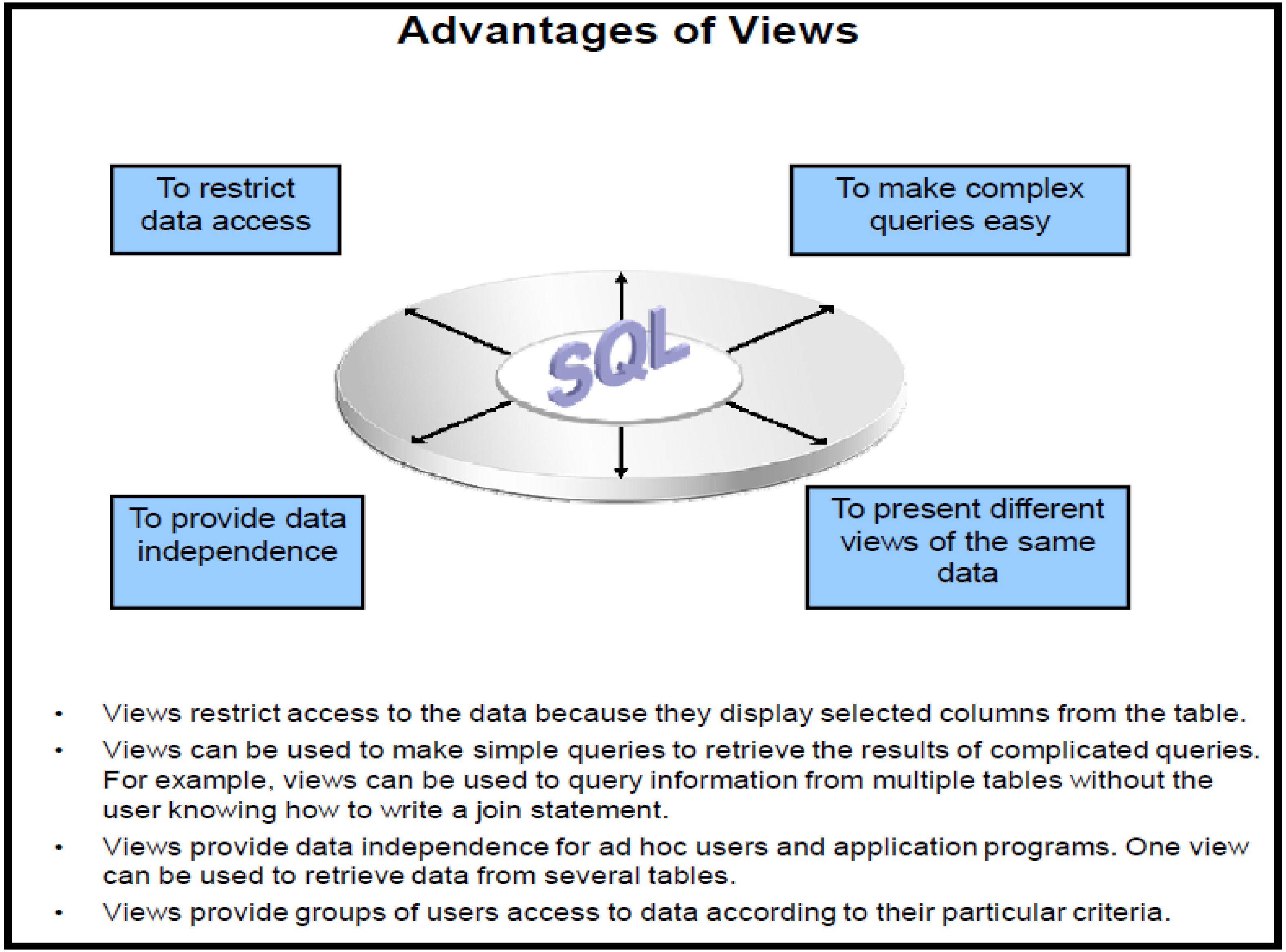
EMPLOYEES table



EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
100	Steven	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000
103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000
104	Bru					ROG	6000
10						ROG	4200
						ACCOUNT	6900
						MAN	5800
						CLERK	3500
						CLERK	3100
						CLERK	2600
						CLERK	2500
						A_MAN	10500
						SA_REP	11000
					R-98	SA_REP	8600
					-MAY-99	SA_REP	7000
					17-SEP-87	AD_ASST	4400
					17-FEB-96	MK_MAN	13000
202	Pat	Fay	PFAY	603.123.6666	17-AUG-97	MK_REP	6000
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94	AC_MGR	12000
206	William	Gietz	WGIETZ	515.123.8181	07-JUN-94	AC_ACCOUNT	8300

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY
100	Steven	King	24000
101	Neena	Kochhar	17000
102	Lex	De Haan	17000
103	Alexander	Hunold	9000
104	Bruce	Ernst	6000

You can present logical subsets or combinations of data by creating views of tables. A view is a schema object , a stored `SELECT` statement based on a table 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 on which a view is based are called *base tables*. The view is stored as a `SELECT` statement in the data dictionary.



Advantages of Views

To restrict data access

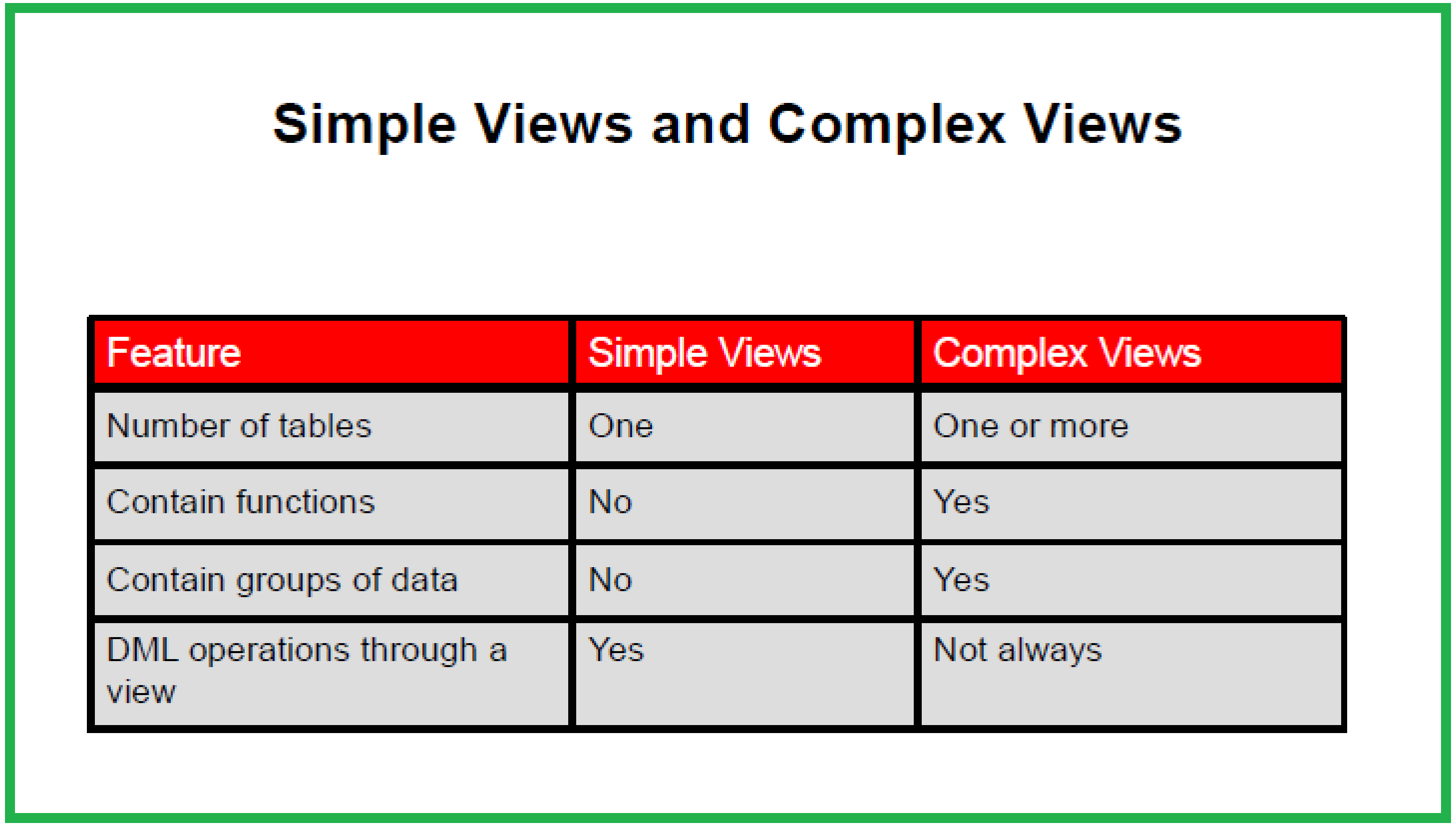
To make complex queries easy

SQL

To provide data independence

To present different views of the same data

- Views restrict access to the data because they display selected columns from the table.
- Views can be used to make simple queries to retrieve the results of complicated queries. For example, views can be used to query information from multiple tables without the user knowing how to write a join statement.
- Views provide data independence for ad hoc users and application programs. One view can be used to retrieve data from several tables.
- Views provide groups of users access to data according to their particular criteria.



Simple Views and Complex Views

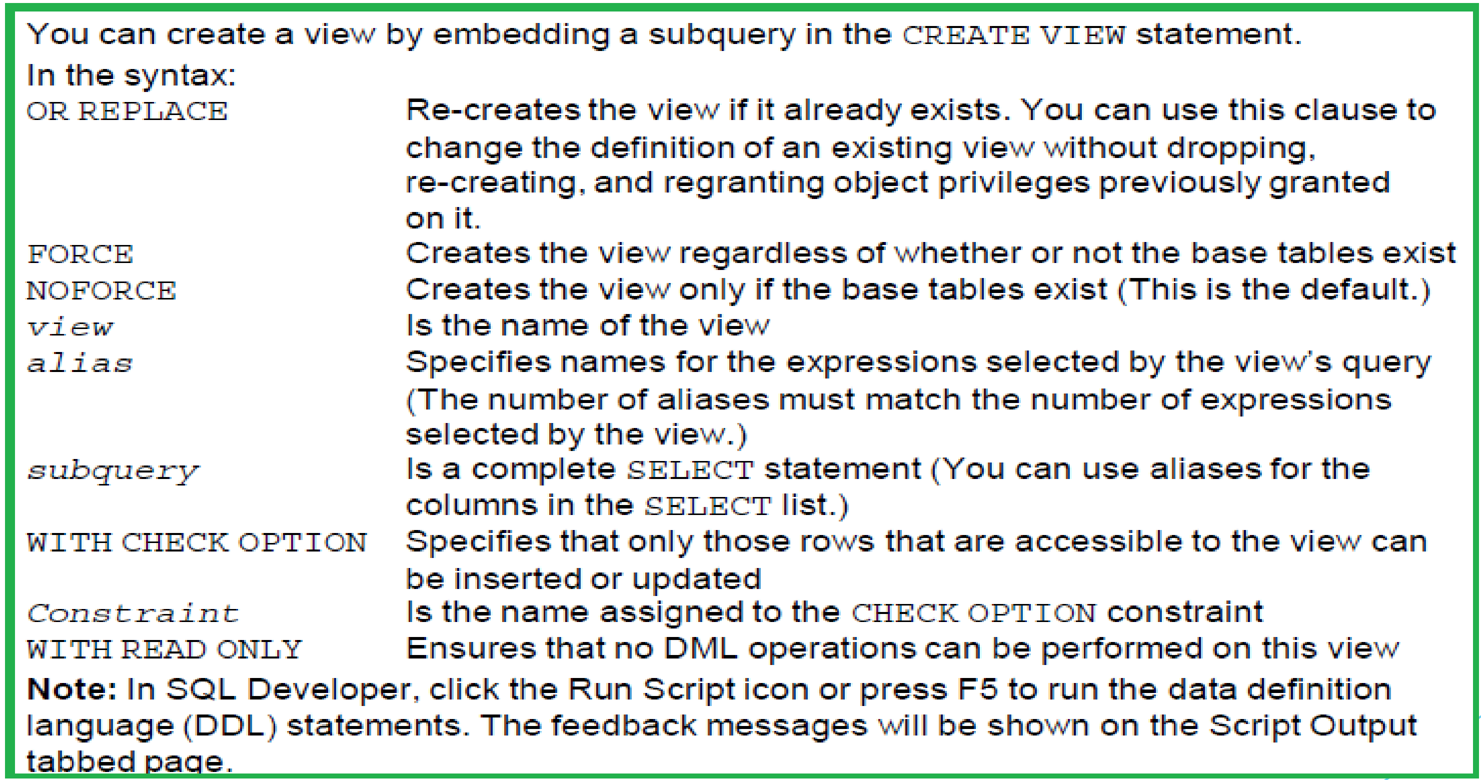
Feature	Simple Views	Complex Views
Number of tables	One	One or more
Contain functions	No	Yes
Contain groups of data	No	Yes
DML operations through a view	Yes	Not always

Creating a View

- You embed a subquery in the CREATE VIEW statement:

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

- The subquery can contain complex SELECT syntax.



You can create a view by embedding a subquery in the CREATE VIEW statement.

In the syntax:

OR REPLACE Re-creates the view if it already exists. You can use this clause to change the definition of an existing view without dropping, re-creating, and regranting object privileges previously granted on it.

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.)

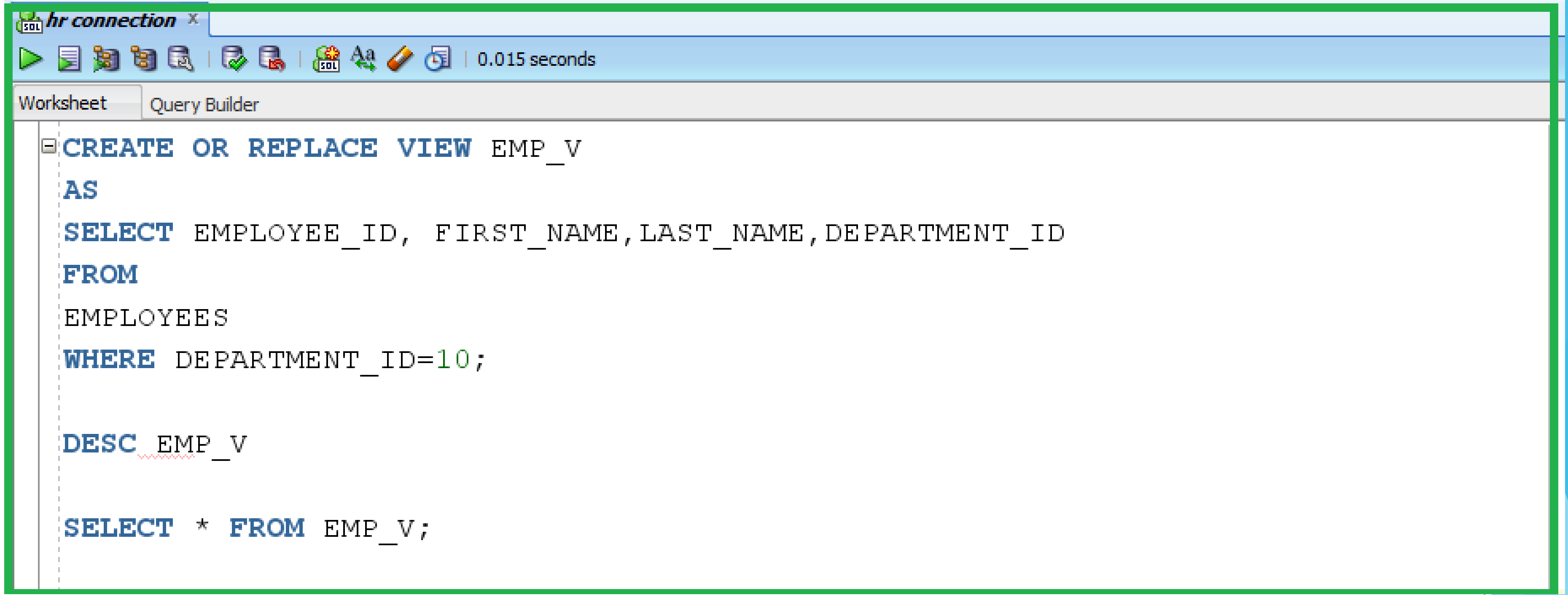
subquery Is a complete SELECT statement (You can use aliases for the columns in the SELECT list.)

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

Note: In SQL Developer, click the Run Script icon or press F5 to run the data definition language (DDL) statements. The feedback messages will be shown on the Script Output tabbed page.

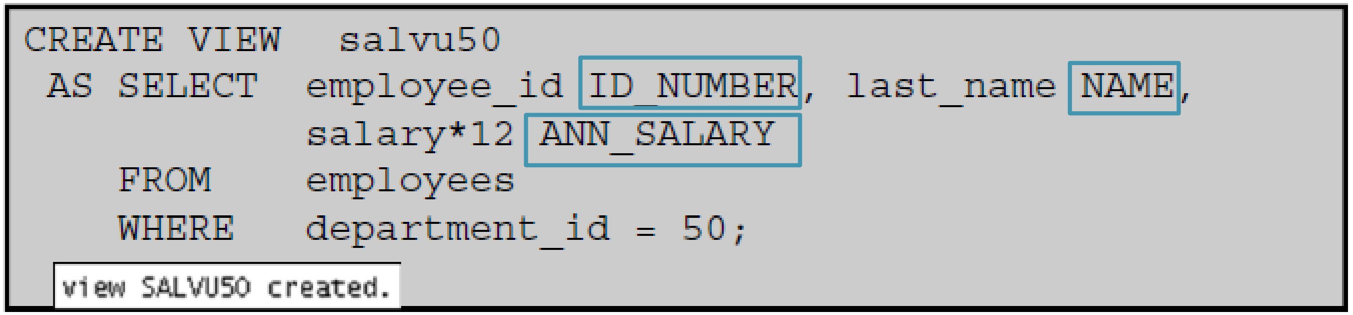


```
CREATE OR REPLACE VIEW EMP_V
AS
SELECT EMPLOYEE_ID, FIRST_NAME,LAST_NAME,DEPARTMENT_ID
FROM
EMPLOYEES
WHERE DEPARTMENT_ID=10;

DESC EMP_V

SELECT * FROM EMP_V;
```

Guidelines

- The subquery that defines a view can contain complex SELECT syntax, including joins, groups, and subqueries.
- If you do not specify a constraint name for the view created with the WITH CHECK OPTION, the system assigns a default name in the SYS_Cn format.
- You can use the OR REPLACE option to change the definition of the view without dropping and re-creating it, or regranting the object privileges previously granted on it.

- Create a view by using column aliases in the subquery:



```
CREATE VIEW   salvu50
 AS SELECT   employee_id ID_NUMBER, last_name NAME,
             salary*12 ANN_SALARY
    FROM     employees
    WHERE    department_id = 50;
```

```
view SALVU50 created.
```

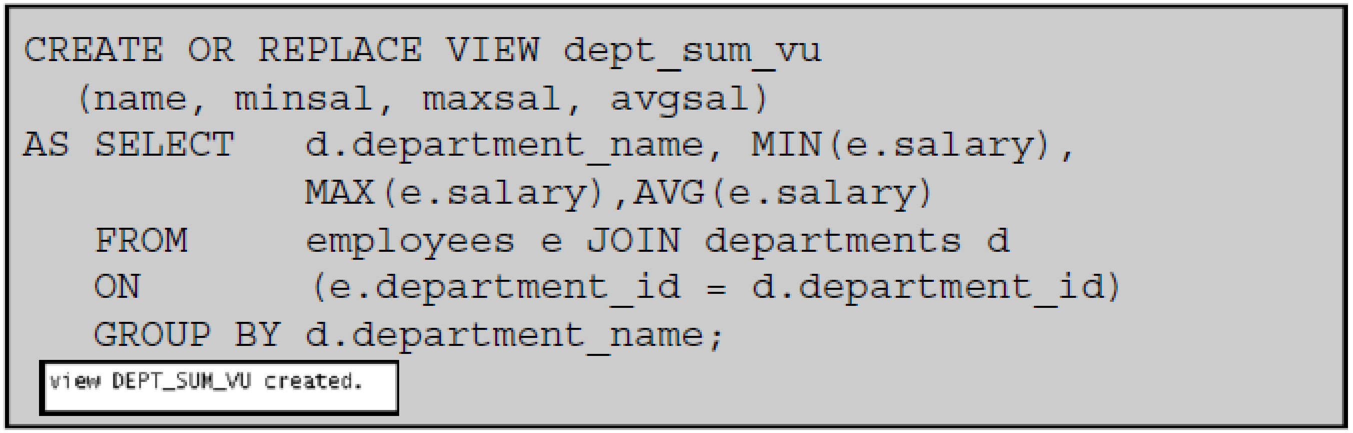
- Select the columns from this view by the given alias names.

Alternatively, you can use an alias after the `CREATE` statement and before the `SELECT` subquery. The number of aliases listed must match the number of expressions selected in the subquery.

```
CREATE OR REPLACE VIEW salvu50 (ID_NUMBER, NAME, ANN_SALARY)
  AS SELECT  employee_id, last_name, salary*12
     FROM    employees
     WHERE   department_id = 50;
```

Creating a Complex View

Create a complex view that contains group functions to display values from two tables:



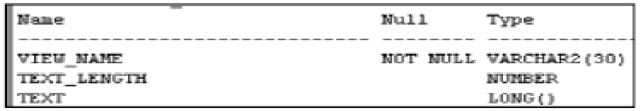
```
CREATE OR REPLACE VIEW dept_sum_vu
  (name, minsal, maxsal, avgsal)
AS SELECT   d.department_name, MIN(e.salary),
            MAX(e.salary),AVG(e.salary)
   FROM     employees e JOIN departments d
   ON       (e.department_id = d.department_id)
   GROUP BY d.department_name;
```

```
view DEPT_SUM_VU created.
```

View Information

1

```
DESCRIBE user_views
```

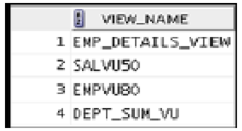


```
Name                           Null     Type
------------------------------ -------- -------------
VIEW_NAME                      NOT NULL VARCHAR2(30)
TEXT_LENGTH                             NUMBER
TEXT                                    LONG()
```

...

2

```
SELECT view_name FROM user_views;
```



	VIEW_NAME
1	EMP_DETAILS_VIEW
2	SALVU50
3	EMPVU80
4	DEPT_SUM_VU

3

```
SELECT text FROM user_views
WHERE view_name = 'EMP_DETAILS_VIEW';
```

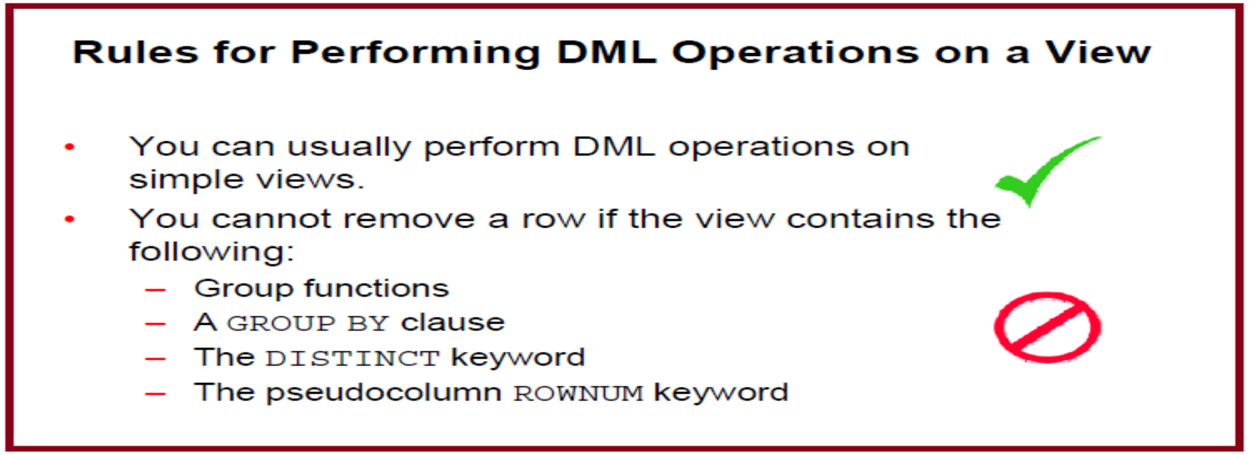
After your view is created, you can query the data dictionary view called USER_VIEWS to see the name of the view and the view definition. The text of the SELECT statement that constitutes your view is stored in a LONG column. The LENGTH column is the number of characters in the SELECT statement. By default, when you select from a LONG column, only the first 80 characters of the column's value are displayed. To see more than 80 characters in SQL*Plus, use the SET LONG command:

```
SET LONG 1000
```

Data Access Using Views

When you access data by using a view, the Oracle Server performs the following operations:

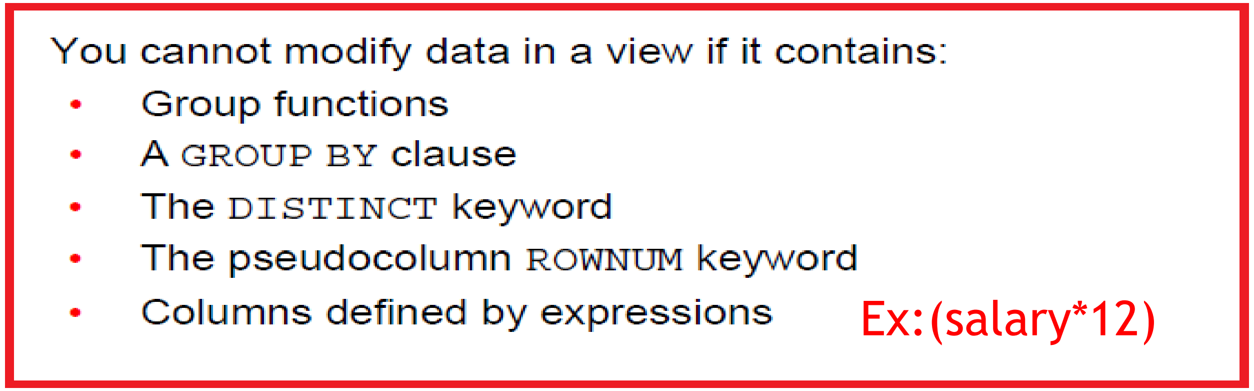
- It retrieves the view definition from the data dictionary table USER_VIEWS.
- It checks access privileges for the view base table.
- It converts the view query into an equivalent operation on the underlying base table or tables. That is, data is retrieved from, or an update is made to, the base tables.



Rules for Performing DML Operations on a View

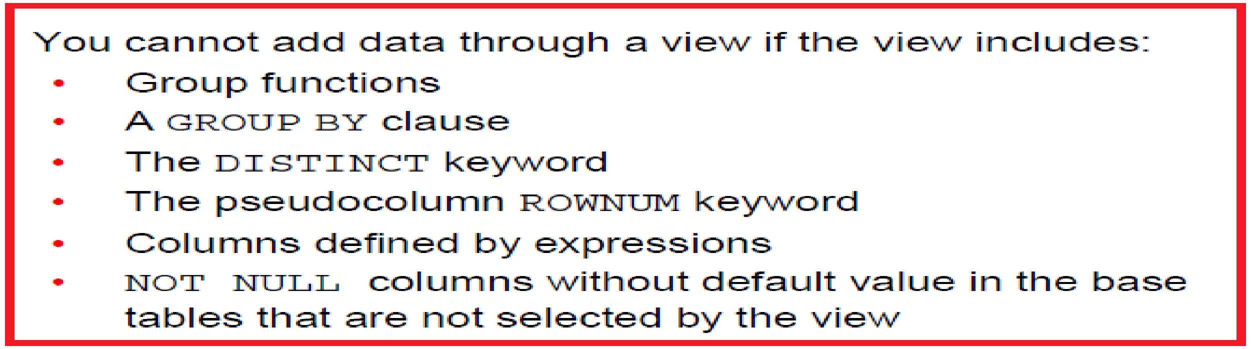
- You can usually perform DML operations on simple views. ✓
- You cannot remove a row if the view contains the following:
  - Group functions
  - A GROUP BY clause
  - The DISTINCT keyword
  - The pseudocolumn ROWNUM keyword

🚫



You cannot modify data in a view if it contains:

- Group functions
- A GROUP BY clause
- The DISTINCT keyword
- The pseudocolumn ROWNUM keyword
- Columns defined by expressions Ex:(salary*12)

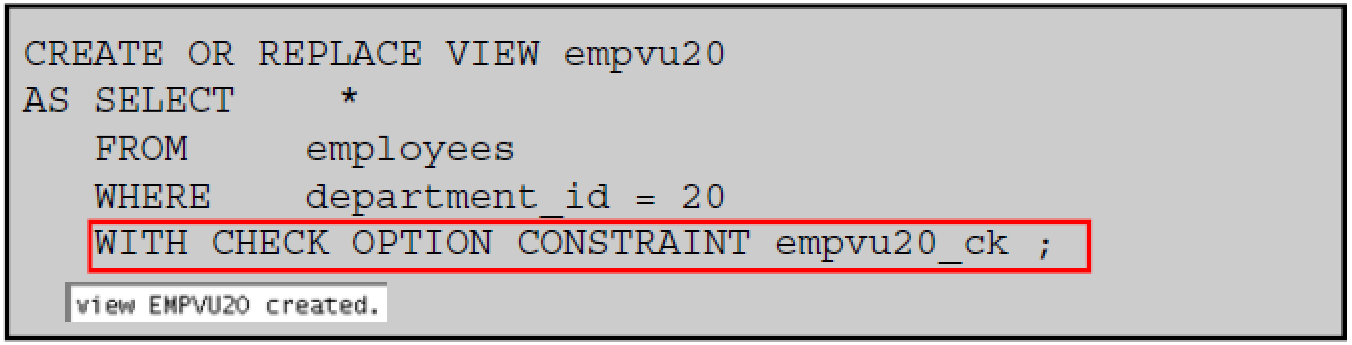


You cannot add data through a view if the view includes:

- Group functions
- A GROUP BY clause
- The DISTINCT keyword
- The pseudocolumn ROWNUM keyword
- Columns defined by expressions
- NOT NULL columns without default value in the base tables that are not selected by the view

Using the WITH CHECK OPTION Clause

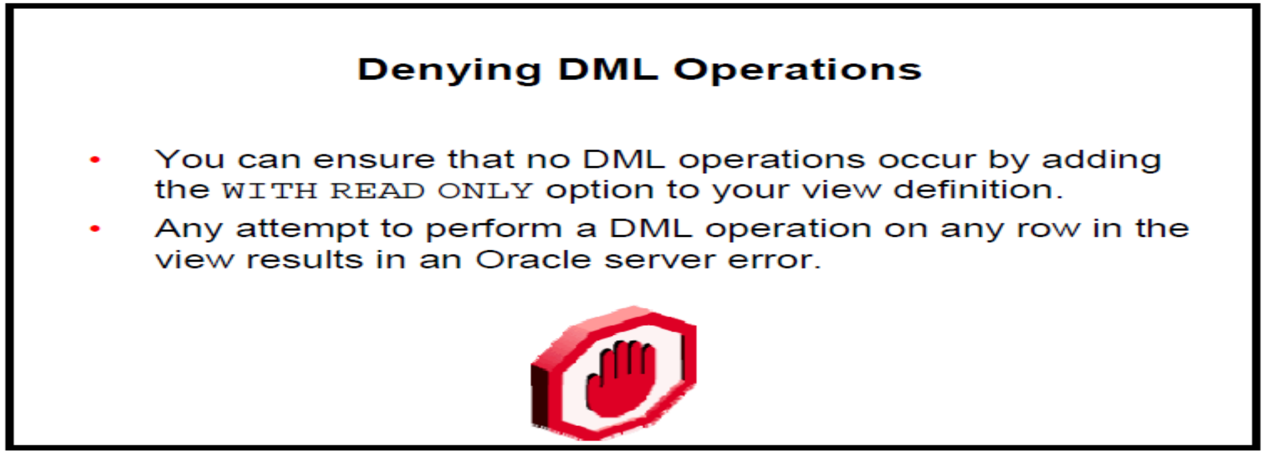
- You can ensure that DML operations performed on the view stay in the domain of the view by using the WITH CHECK OPTION clause:



```
CREATE OR REPLACE VIEW empvu20
AS SELECT     *
   FROM      employees
   WHERE     department_id = 20
   WITH CHECK OPTION CONSTRAINT empvu20_ck ;
```

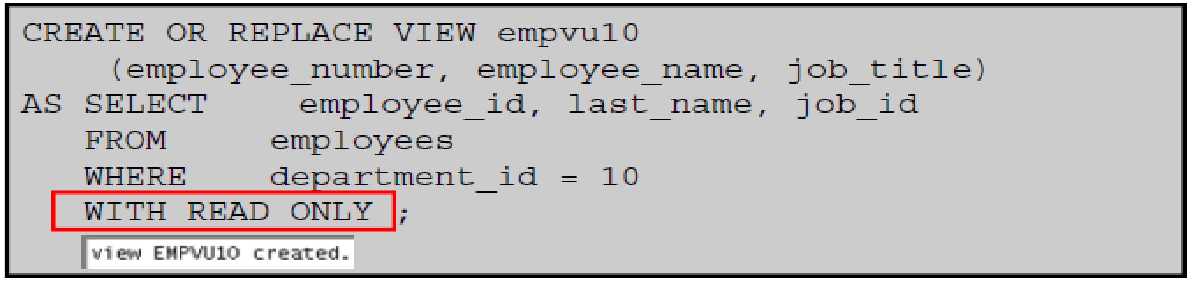
```
view EMPVU20 created.
```

- Any attempt to INSERT a row with a department_id other than 20 or to UPDATE the department number for any row in the view fails because it violates the WITH CHECK OPTION constraint.



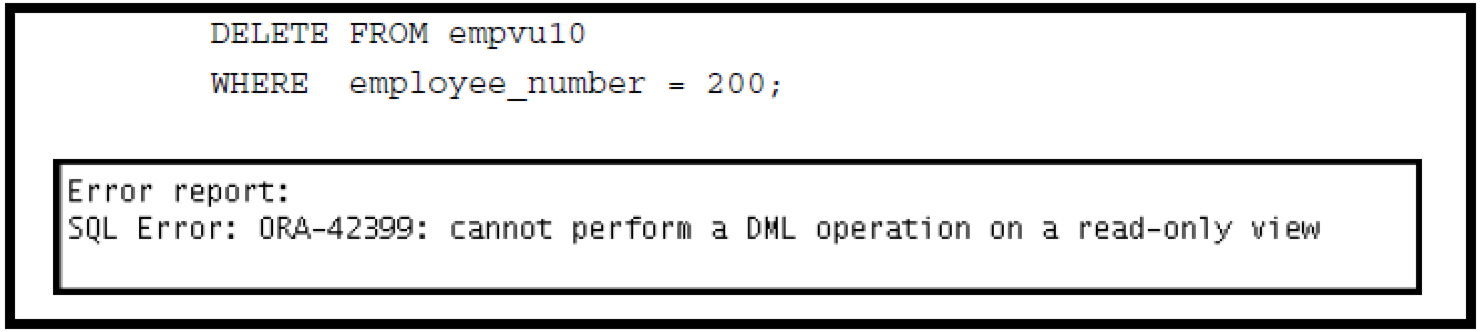
Denying DML Operations

- You can ensure that no DML operations occur by adding the WITH READ ONLY option to your view definition.
- Any attempt to perform a DML operation on any row in the view results in an Oracle server error.



```
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;
```

```
view EMPVU10 created.
```



```
DELETE FROM empvu10
WHERE  employee_number = 200;
```

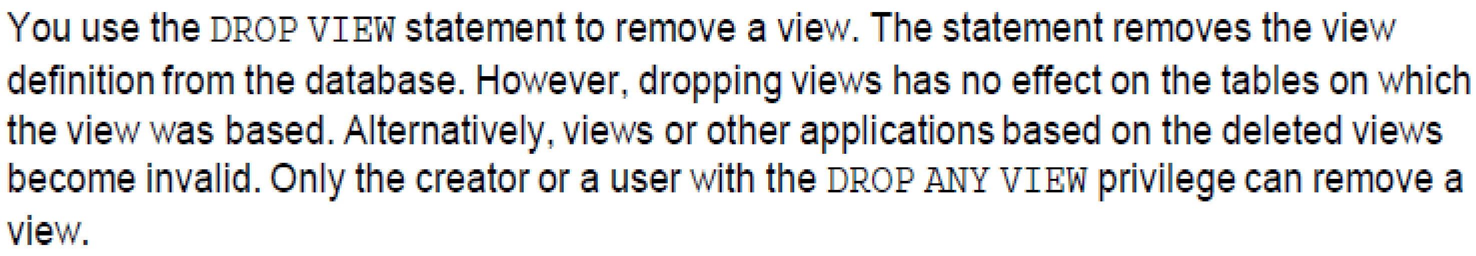
```
Error report:
SQL Error: ORA-42399: cannot perform a DML operation on a read-only view
```

Removing a View

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

```
DROP VIEW view;
```

```
DROP VIEW empvu80;
```



You use the DROP VIEW statement to remove a view. The statement removes the view definition from the database. However, dropping views has no effect on the tables on which the view was based. Alternatively, views or other applications based on the deleted views become invalid. Only the creator or a user with the DROP ANY VIEW privilege can remove a view.

Thank You