VIEWS:(18-11-2024)
======

- it is a subset / virtual / logical table of a base table.
- view does not store data / information.
- the main advantage of view is providing security for data.there are two levels of security.

i) colum level:
============

- In this level we are hiding columns.

ii) row level:
===========

- In this level we are hiding rows.

- whenever we perform DML operations on a view internally those operations are executed on a base table and reflected in view table.

Types of Views:
==============

- there are two types of views.
  1. simple view
  2. complex view

1. simple view:
=============

- when we created a view for accessing the required data from a single base table is known as simple view.
- by default simple view is allowed DML operations on a base table.

syntax:
=======

```
create view <view name> as <select query>;
```

EX:

create a view to access the data from DEPT table?

```
CREATE VIEW V1 AS SELECT * FROM DEPT;
```

TESTING:
=========

```
INSERT INTO V1 VALUES(50,'DBA','HYD');
UPDATE V1 SET LOC='PUNE' WHERE DEPTNO=50;
DELETE FROM V1 WHERE DEPTNO=50;
SELECT * FROM V1;
```

EX:

create a view to access EMPNO,ENAME,SALARY from emp table?

```
CREATE VIEW V2 AS SELECT EMPNO,ENAME,SAL FROM EMP;
```

TESTING:
=========

```
INSERT INTO V2 VALUES(1122,'YUVIN',8500);
SELECT * FROM V2;
```

EX:
create a view to access employees details from EMP table who are working under deptno is 20?

```
CREATE VIEW V3 AS SELECT * FROM EMP WHERE DEPTNO=20;
```

TESTING:
=========

```
SELECT * FROM V3;
```

EX:
create a view to display and accept employees details whose employee salary is 3000?

```
CREATE VIEW V4 AS SELECT * FROM EMP WHERE SAL=3000;
```

TESTING:

```
INSERT INTO V4 VALUES(1122,'YUVIN','HR',7566,'2024-11-18',2500,NULL,10);----ALLOWED
SELECT * FROM V4;
SELECT * FROM EMP;
```

- In the above example we are displaying employees whose salary is 3000 but we are not restricted employees whose salary is not 3000.
- To overcome the above problem we must use an option is "WITH CHECK OPTION" statement while creating a view object.

WITH CHECK OPTION:
===================

- this statement is used to restrict data on base table through a view object.

EX:

```
CREATE VIEW V5 AS SELECT * FROM EMP WHERE SAL=3000 WITH CHECK OPTION;
```

TESTING:
=========

```
INSERT INTO V5
VALUES(1123,'BHUVIN','DEVELOPER',7698,'2024-11-12',4000,NULL,30);----NOT ALLOWED
INSERT INTO V5
VALUES(1123,'BHUVIN','DEVELOPER',7698,'2024-11-12',3000,NULL,30);----ALLOWED

SELECT * FROM EMP;
SELECT * FROM V5;
```

19-11-2024:
===========

WITH ENCRYPTION:
=================

- this statement is used to hide a view definition from users.
- this statement should use at header level of a view definition.

syntax:
=======

```
create view <view name> with encryption as <select query>;
```

EX:

```
CREATE VIEW V6 WITH ENCRYPTION
AS
SELECT * FROM DEPT;
```

How to see the definition of a view in sqlserver:
==

syntax:
=======

```
SELECT TEXT FROM SYSCOMMENTS WHERE ID=OBJECT_ID('view name');
```

Here,

SYSCOMMENTS is a system defined table.
TEXT,ID are colums of syscomment table.
OBJECT_ID is a method name.

EX:

```
SELECT TEXT FROM SYSCOMMENTS WHERE id=OBJECT_ID('V6');
```

OUTPUT:
========

```
TEXT
======
NULL
```

Alerting a VIEW in sqlserver:
=========================

syntax:
=======

```
ALTER VIEW <VIEW NAME> AS <SELECT QUERY>;
```

EX:

```
ALTER VIEW V6
```

```
AS
SELECT * FROM DEPT;
```

EX:

```
SELECT TEXT FROM SYSCOMMENTS WHERE id=OBJECT_ID('V6');
```

OUTPUT:

```
========
TEXT
======
CREATE VIEW V6   AS  SELECT * FROM DEPT
```

WITH SCHEMABINDING:
====================

- this statement is used to restrict to drop / delete a main table from database.
- this statement should apply at heder level while creating a view object.
- when we apply "with schemabinding" option in the view then we must follow the following two conditions.
    - i) should not use " * " in SELECT query.
    - ii) use a table name in two parts wise i.e DBO.<object name>;

syntax:

```
======
CREATE VIEW <VIEW NAME> WITH SCHEMABINDING AS <SELECT QUERY>;
```

EX:

```
CREATE VIEW V7 WITH SCHEMABINDING
AS
SELECT DEPTNO,DNAME,LOC FROM DBO.DEPT
```

TESTING:

```
=========
DROP TABLE DEPT;----------> NOT ALLOWED
```

2) COMPLEX VIEW:
================

- when we created a view based on :
    - > multiple tables.
    - > by using "group by" clause.
    - > by using "distinct" keyword.
    - > by using "having" clause.
    - > by using "set operators".
    - > by using joins.

```
        > by using subquery.
        > by using aggregative functions then we called as complex view.
    - by default complex views are not allowed DML operations on base table.so that these views
    are called as "read only views" in database.
```

syntax:

=======

```
create view <view name> as <select  query>;
```

EX:

```
CREATE VIEW V8
AS
SELECT * FROM EMP_HYD
UNION
SELECT * FROM EMP_MUMBAI
```

TESTING:

=========

```
SELECT * FROM V8; -------------------> ALLOWED
INSERT INTO V8 VALUES(1025,'ADAMS',63000);-------NOT ALLOWED
UPDATE V8 SET SALARY=11000 WHERE EID=1023;---NOT ALLOWED
DELETE FROM V8 WHERE EID=1021;-------NOT ALLOWED
```

EX:

```
CREATE VIEW V9 AS SELECT DEPTNO,SUM(SAL) AS SUM_OF_SALARY FROM EMP
GROUP BY DEPTNO;
```

TESTING:

=========

```
SELECT * FROM V9; (before updating)
UPDATE EMP SET SAL=SAL+1000 WHERE DEPTNO=10;
SELECT * FROM V9; (after updating)
```

How to drop a view:

=================

syntax:

=======

```
DROP VIEW <VIEW NAME>;
```

EX:

```
DROP VIEW V1;
```