View 정의

[정의] a logical table based on a table or another view, A view contains no data of its own, but is rather like a "window" through which data from table can be viewed or changed. the view is stored as a SELECT statement in the data dictionary.

Stored Query (SQL 문장만 저장된다)

oracle store view's definition as a text
therefore not space,not indexed

- 질의를 재작성(Rewrite)하여 수행 하며, 데이터를 가지고 있지 않지만 테이블의 역할을 동일하게 수행하기 때문에 가상 테이블(Virtual Table)로 불리기도 한다.

[종류]

View
Inline View
Partition VIEW
Object View
Materialized View

View merge (in all possible case)

```
[VIEW 정의)] CREATE   VIEW   V_DEPTEMP
             AS   SELECT   E.EMPNO,E.ENAME,E.SAL,D.DNAME
                  FROM    EMP   E, DEPT D   WHERE    E.DEPTNO = D.DEPTNO;
[VIEW 질의] SELECT   *   FROM V_DEPTEMP WHERE   DNAME   =   'SALES'
[VIEW 처리] SELECT   E.EMPNO,E.ENAME,E.SAL,D.DEPTNO,D.DNAME
            FROM     EMP   E,   DEPT D
            WHERE    E.DEPTNO = D.DEPTNO   AND   D.DNAME = 'SALES';
```

Oject Dependency

BASE OBJECT(TABLE or VIEW)의 DROP 시 VIEW 의 상태는?

```
EX)  CREATE TABLE BASE_TBL(EMPNO_NEW,ENAME_NEW)
      AS      SELECT   EMPNO,ENAME   FROM EMP;
      CREATE   VIEW   DEPEN   AS   SELECT EMPNO,ENAME FROM BASE_TBL;

     SELECT OBJECT_NAME,STATUS   FROM USER_OBJECTS WHERE OBJECT_NAME ='DEPEN';
     DROP   TABLE   BASE_TBL;
```

SELECT OBJECT_NAME,STATUS FROM USER_OBJECTS WHERE OBJECT_NAME ='DEPEN';

DROP 된 BASE OBJECT(TABLE or VIEW)가 다시 생성된후 VIEW 의 상태는?

CREATE TABLE BASE_TBL(EMPNO_NEW,ENAME_NEW,DEPTNO_NEW)
AS SELECT EMPNO,ENAME,DEPTNO FROM EMP;

SELECT OBJECT_NAME,STATUS FROM USER_OBJECTS WHERE OBJECT_NAME ='DEPEN';

SELECT * FROM DEPEN;

SELECT OBJECT_NAME,STATUS FROM USER_OBJECTS WHERE OBJECT_NAME ='DEPEN';

DROP 된 BASE OBJECT(TABLE or VIEW)가 VIEW 의 정의와 다르게 다시 생성된후 VIEW 의 상태는?

View 용도

① **보안성(RESTRICT DATABASE ACCESS)** ~ 보여지는 데이타를 선택하여 제한 한다.

EX) CREATE VIEW RESTRICT_SELECT(EMPNO_NEW,ENAME_NEW,DEPTNO_NEW)
AS SELECT EMPNO,ENAME,DEPTNO FROM EMP WHERE SAL > 1500;

SELECT * FROM RESTRICT_SELECT; // BASE-DATA 의 COLUMN,ROW 를 제한

② **편리성(SIMPLIFY QUERIES)** ~ 복잡하게 JOIN 된 VIEW 를 단순하게 조회 할수있다

EX) CREATE VIEW SIMPLE_VIEW
AS SELECT EMP.EMPNO,EMP.ENAME,DEPT.DNAME,SALGRADE.GRADE AS GD
FROM EMP,DEPT,SALGRADE
WHERE EMP.DEPTNO = DEPT.DEPTNO AND
EMP.SAL BETWEEN SALGRADE.LOSAL AND SALGRADE.HISAL;

DESC SIMPLE_VIEW

SELECT * FROM SIMPLE_VIEW;

* TABLE(VIRTUAL TABLE) 처럼 NOT NULL, DATA TYPE 까지

③ **독립성(DATA INDEPENDENCE) ~ 테이블구조 변경시 View 를 사용하는 응용 프로그램을 변경하지 않아도 된다.**

(ISOLATE APPLICATIONS FROM CHANGES IN DEFINITIONS OF BASE TABLES)

EX) CREATE TABLE BASE_TBL(EMPNO_NEW,ENAME_NEW)
AS SELECT EMPNO,ENAME FROM EMP;

CREATE VIEW IND_VIEW AS SELECT * FROM BASE_TBL;

SELECT * FROM IND_VIEW;

ALTER TABLE BASE_TBL ADD(NEW_COL DATE); // BASE TABLE 변경.

SELECT * FROM IND_VIEW; // INDEPENDENCE OF TABLE CHANGE

④ **DIFFERENT APPEARANCES FOR THE SAME DATA**

동일한 데이타도 USER, 시간에 따라 다른 데이타 결과를 보여줄수 있다

EX) DBA_TABLES, USER_TABLE

EX) CONNECT SYSTEM/ MANAGER

```
CREATE TABLE   DIFF_RETURN_TBL
AS   SELECT OWNER,TABLE_NAME FROM DBA_TABLES;
SELECT  *   FROM   DIFF_RETURN_TBL;  // DATA 의 갯수 확인.

CREATE OR REPLACE VIEW   DIFF_RETURN_VIEW
AS   SELECT  *  FROM   DIFF_RETURN_TBL   WHERE OWNER = USER;

SELECT   *  FROM   DIFF_RETURN_TBL;

GRANT SELECT ON DIFF_RETURN_VIEW   TO PUBLIC;                    //모든사용자에게 권한부여

CONNECT SCOTT/SCOTT
SELECT * FROM DIFF_RETURN_VIEW;
```

Syntax

[구문] CREATE [OR REPLACE] [FORCE | NOFORCE] VIEW
view_name[(column-alias1,column-alias2.....)]
AS Subquery
[WITH CHECK OPTION [CONSTRAINT constraint]]
[WITH READONLY]

OR REPLACE :VIEW 에는 MODIFY 가 없다, VIEW 의 정의를 변경하려고 할때는
DROP → RE-CREATE :VIEW DROP 시 부여한 object-privilege 도 삭제된다.
OR REPLACE :이미 할당된 object-privilege 가 계속 유효하다.

EX) // 차이점?
```
DROP    VIEW   DATA_IND;
CREATE  VIEW   DATA_IND(ID_NUM,NAME_CHAR)
          AS   SELECT   ID,NAME   FROM   TMP;

CREATE   OR REPLACE   VIEW   DATA_IND   AS   SELECT   ID,NAME   FROM   TMP;
```

FORCE : BASE TABLE 의 존재 여부에 상관 없이 VIEW 를 만든다(VIEW 를 먼저만든다).

EX) CREATE VIEW FORCE_VIEW AS SELECT * FROM FORCE_TABLE;

//결과는 ? USER_VIEWS : VIEW_NAME,STATUS

CREATE FORCE VIEW FORCE_VIEW AS SELECT * FROM FORCE_TABLE;

//결과는 ?

USER_VIEWS : VIEW_NAME,STATUS USER_OBJECTS:OBJECT_NAME,STATUS

NOFORCE: DEFAULT, BASE TABLE 이 존재해야만 VIEW 를 만들수 있다.

WITH CHECK OPTION: VIEW 에 의해서 접근 가능한 ROW 만이 INSERT,UPDATE 될수있다.

CONSTRAINT constraint : WITH CHECK OPTION 의 CONSTRAINT 명

CONSTRAINT 명을 명기 하지 않는 경우 DEFAULT NAME:SYS_Cn

EX) CREATE VIEW TEST_CHK
AS SELECT * FROM EMP WHERE DEPTNO = 10
WITH CHECK OPTION CONSTRAINT TEST_CHK_DEPT_10;

WITH READONLY : DML 이 VIEW 에 사용되지 못하도록 한다, 읽기 전용.

SIMPLE-VIEW vs COMPLEX_VIEW

내용	SIMPLE VIEW	COMPLEX VIEW
TABLE 의 개수	ONLY ONE	1 개이상
함수의 사용여부	NO	YES
데이타 그룹여부	NO	YES
DML 을사용가능여부	YES	가능한 경우도 있음

* ORACLE 7.3 이후부터 JOIN 된 VIEW 에서도 경우에따라 UPDATE 가 가능한 VIEW 있음

COMPLEX VIEW

```
EX) CREATE VIEW    COMPLEX_JVIEW     //  JOIN 을 사용한  COMPLEX  VIEW
      AS  SELECT EMP.EMPNO,EMP.ENAME,DEPT.DNAME,SALGRADE.GRADE  AS  GD
          FROM     EMP,DEPT,SALGRADE
          WHERE  EMP.DEPTNO = DEPT.DEPTNO  AND
                       EMP.SAL BETWEEN SALGRADE.LOSAL AND SALGRADE.HISAL;

   CREATE OR REPLACE VIEW  COMPLEX_FVIEW // FUNCTION 을이용한 COMPLEX VIEW
   AS
   SELECT  DEPTNO,EMPNO,
   DECODE(DEPTNO,10,'ACCOUNTING',20,'RESEARCH',
                 30,'SALES',40,'OPERATIONS') DNAME
   FROM EMP;

  CREATE OR REPLACE VIEW COMPLEX_GVIEW  // DATA GROUP  COMPLEX VIEW
  AS     SELECT DEPTNO,MIN(SAL) as Min_Sal,MAX(SAL) "Max Salary"  FROM EMP  .
         GROUP BY DEPTNO  ORDER BY Min_Sal;                    //?수정하십시요
```

View with DML , with Check , with read only

RULES FOR PERFORMING DML OPERATIONS ON A VIEW

YOU CAN PERFORM DML OPERATION ON SIMPLE VIEWS

DELETE 시 제한

① JOIN CONDITION
② GROUP FUNCTIONS, GROUP BY
③ DISTINCT

UPDATE 시 제한

① 위의 DELETE 시 제한 사항.
② COLUMNS DEFINED BY EXPRESSIONS // (SAL*12 + 300) AS ANNUAL_SAL
③ ROWNUM PSEUDO COLUMN 을 포함한 경우.

INSERT 시

① 위의 DELETE,UPDATE 시 제한 사항.

② ANY NOT NULL COLUMNS NOT SELECTED BY THE VIEW

WITH CHECK OPTION

VIEW 를 구성하는 조건에 맞는 데이타가 변경(INSERT,UPDATE)되지 못하도록 한다

EX) CREATE OR REPLACE VIEW EMP_VIEW
AS SELECT * FROM EMP_NEW WHERE DEPTNO = 10;
UPDATE EMP_VIEW SET DEPTNO = 20 ; // 결과는,SELECT,ROLLBACK;
INSERT INTO EMP_VIEW(EMPNO,ENAME,DEPTNO) VALUES(9999,'VIEW_TEST',10)
INSERT INTO EMP_VIEW(EMPNO,ENAME,DEPTNO) VALUES(9999,'VIEW_TEST',20)
DELETE FROM EMP_VIEW WHERE DEPTNO = 10; //결과는?.

// OR REPLACE 로 기존의 EMP_VIEW 를 DROP 하지 않고 다시 생성한다.
CREATE OR REPLACE VIEW EMP_VIEW
AS SELECT * FROM EMP_NEW A WHERE DEPTNO = 10
WITH CHECK OPTION CONSTRAINT EMP_VIEW_DEPTNO_10;

UPDATE EMP_VIEW SET DEPTNO = 20 ; // 결과는,SELECT,ROLLBACK;
INSERT INTO EMP_VIEW(EMPNO,ENAME,DEPTNO) VALUES(9999,'VIEW_TEST',10);
INSERT INTO EMP_VIEW(EMPNO,ENAME,DEPTNO) VALUES(9999,'VIEW_TEST',20);
DELETE FROM EMP_VIEW WHERE DEPTNO = 10; // 결과는??

THE VIEW CAN ONLY SEE DEPTNO 10 , AND DOES NOT ALLOW THE DEPARTMENT

NUMBER FOR THOSE EMPLOYEE TO BE CHANGED THROUGH THE VIEW

WITH READ ONLY ~ DML OPERATION 이 VIEW 에서 발생하지 않도록한다.

EX) CREATE OR REPLACE VIEW EMP_VIEW
AS SELECT * FROM EMP_NEW A WHERE DEPTNO = 10
WITH READ ONLY;

// 결과는 ?
UPDATE EMP_VIEW SET DEPTNO = 10 ;
INSERT INTO EMP_VIEW(EMPNO,ENAME,DEPTNO) VALUES(9999,'VIEW_TEST',10)
DELETE FROM EMP_VIEW WHERE DEPTNO = 10;

과제 1)생성된 VIEW 에대해 USER_VIEWS 를 가지고 VIEW 의 NAME 과 DEFINITION 를 찾아보십시요.

과제 2)기 생성된 EMP_VIEW 를 BASE TABLE 로 하여 SAL 가 2000 인 조건의 VIEW 를 생성하십시요

과제 3) 과제 2 에서 생성한 VIEW 를 Am 09:30 분부터 Pm 14:56 까지만 SELECT 되도록 바꾸시오.

Drop View

VIEW 자체는 DATA DICTIONARY 에 VIEW 의 정의만이 SQL 형태로 저장이 되어 있기 때문에
DROP VIEW 는 DATA DICTIONARY 에서 단지 VIEW 의 정의만을 삭제하는것이다. VIEW 를 DROP 하는것은
TABLE 정의와 TABLE 데이타에는 아무런 영향이 없다.

구문) DROP VIEW view_name;

EX) DROP VIEW EMP_VIEW;

과제 1) SCRIPT FILE 로 DROP SQL 문을 자동으로 생성하여 예제로 작성한 모든 VIEW 를 DROP 하십시요.

과제 2) VIEW 의 이름을 입력받아 해당 VIEW 를 삭제하는 SCRIPT FILE 을 작성하십시오.