

## 12. 뷰의 생성과 활용

컴퓨터공학부

김은경

## [학습 목표]

1. 뷰 사용의 장점을 나열할 수 있다.
2. CREATE VIEW 명령을 사용하여 뷰를 생성할 수 있다.
3. CREATE OR REPLACE VIEW 명령을 사용하여 뷰를 수정할 수 있다.
4. 테이블 대신 뷰를 통해서 데이터를 검색할 수 있다.
5. 뷰에서의 DML 연산의 수행 규칙을 설명할 수 있다.

# 1. 뷰 개요

## ★ 뷰(View)란?

- 하나 이상의 테이블의 데이터의 부분 집합으로 구성되는 논리적인 테이블

## ★ 뷰의 특징

- 테이블 뿐만 아니라 다른 뷰를 기초로도 생성할 수 있음
- 뷰 자체는 데이터를 직접 포함하지 않지만, 창문 역할을 하는 뷰를 통해서 데이터의 검색 및 수정이 가능함
- 열 별칭을 사용해서 생성된 뷰에 대해서는 열 별칭을 사용한 조작만 가능함

# 1. 뷰 개요

## ★ 뷰 사용의 장점

- ① 보안을 위해서 DB에 대한 접근을 제한할 수 있음
  - 사용자는 특정 테이블의 데이터 가운데 뷰로 정의된 특정 부분만을 보게 됨
- ② 복잡한 질의를 단순한 질의로 변환할 수 있음
  - 다중 테이블에서 뷰를 생성하면 테이블 조인이 불필요하게 됨
  - 즉, 주로 사용하는 정보만을 대상으로 데이터 조작을 수행할 수 있음
- ③ 데이터 독립성을 허용함
  - 테이블이 변경되어도 뷰는 그대로 유지할 수 있으므로, 임시 사용자와 응용 프로그램에 대한 데이터 독립성을 제공할 수 있음
- ④ 동일한 데이터에 대해서 다른 뷰를 생성할 수 있음
  - 조건에 따라 데이터에 접근하는 사용자 그룹 분류해서, 각각 동일한 테이블에 대한 다른 뷰를 기초로 데이터 조작을 할 수 있게 함

# 1. 뷰 개요

## ★ 뷰의 종류

### ① 단순 뷰

- 오직 하나의 테이블만을 기초로 생성된 뷰를 의미함
- 표현식 등에 의해 데이터가 조작된 경우를 제외하면, 뷰를 통한 모든 DML 연산의 수행이 가능함

### ② 복합 뷰

- 다중 테이블을 기초로 생성된 뷰를 의미함
- 데이터 그룹핑 또는 그룹 함수를 사용해서 뷰를 생성할 수 있음
- 뷰를 통한 모든 DML이 항상 가능한 것은 아님

## 2. 뷰 생성

### ★ 뷰 생성 방법

- CREATE VIEW 명령문에서 서브쿼리를 이용해서 생성하고, 뷰가 생성된 다음 뷰 이름과 뷰 정의는 데이터 사전의 USER\_VIEWS 테이블에 저장됨

```
CREATE [FORCE | NOFORCE] VIEW 뷰이름 [(열별칭1[, 열별칭2, ...])]
AS 서브쿼리
[WITH CHECK OPTION [CONSTRAINT 제약이름]]
[WITH READ ONLY];
```

- 열별칭 : 서브쿼리에 의해 선택된 열이나 표현식에 대한 별칭을 지정함
- FORCE : 기본 테이블의 존재 여부와 무관하게 뷰를 생성함
- NOFORCE : 기본 테이블이 존재할 때만 뷰를 생성함
- 서브쿼리 : 뷰에 포함될 데이터를 검색하는 SELECT 문을 작성함
- WITH CHECK OPTION : 뷰에 의해 접근 가능한 행만 삽입 또는 수정될 수 있음을 명시함
- WITH READ ONLY : 뷰에 대해서 SELECT 만 가능하고, 다른 DML 연산은 불가능함을 명시함

## 2. 뷰 생성

### ✓ 단순 뷰 생성 예제

- 200번 부서의 모든 사원의 번호와 이름, 입사일을 포함하는 뷰인 empview200 생성하기

```
CREATE VIEW empview200  
AS SELECT emp_id, emp_name, hiredate  
FROM emp  
WHERE dept_id = 200;
```

View created.

```
SELECT *  
FROM empview200;
```

EMP_ID	EMP_NAME	HIREDATE
6200	박철수	17/12/17
6361	김철수	20/06/09
7878	백기수	20/12/09
7933	김철수	21/01/23
7631	박은미	21/03/01

## 2. 뷰 생성

### ✓ 복합 뷰 생성 예제

- 300번 부서의 모든 사원의 번호와 이름, 업무, 근무위치를 포함하는 뷰인 empview300 생성하기

```
CREATE VIEW empview300
AS SELECT emp_id, emp_name, job, dept_loc
FROM emp, dept
WHERE emp.dept_id = 300
AND emp.dept_id = dept.dept_id;
```

View created.

```
SELECT *
FROM empview300;
```

EMP_ID	EMP_NAME	JOB	DEPT_LOC
6351	진대준	부장	서울
7910	이영희	경리	서울
7920	김마리아	사무직	서울
7696	이혜성	사무직	서울



## 2. 뷰 생성

### ✓ 서브쿼리에서 열 별칭을 사용한 뷰 생성 예제

- 100번 부서의 모든 사원의 사원번호와 이름, 급여만을 포함하는 뷰를 생성 하되, 각각의 열 이름이 employee\_id, employee\_name, salary가 되게 하기

```
CREATE VIEW  salview100
AS SELECT  emp_id employee_id, emp_name employee_name,
           sal salary
FROM      emp
WHERE     dept_id = 100;
```

View created.

```
SELECT  employee_id
FROM    salview100;
```

EMPLOYEE_ID	
	6311
	7872

## 2. 뷰 생성

### ✓ 열 별칭을 사용한 뷰 생성 예제

- 400번 부서의 모든 사원의 사원번호와 급여만을 포함하는 뷰를 생성하되, 각각의 열 이름이 employee\_id와 salary가 되게 하기

```
CREATE VIEW salview400(employee_id, salary) AS SELECT emp_id, sal
FROM emp
WHERE dept_id = 400;
```

1행에 오류:  
ORA-00904: 열명이 부적합합니다

View created.

```
SELECT employee_id, salary
FROM salview400;
```

EMPLOYEE_ID	SALARY
7489	3700
7522	4520
6321	7000
7854	3500

### [비교]

```
SELECT emp_id, sal
FROM salview400;
```

```
SELECT emp_id, sal
      *
ERROR at line 1:
ORA-00904: "SAL": invalid identifier
```

열 이름이 부적절함

## 2. 뷰 생성

### ✓ 복합 뷰 생성 예제

- 부서이름과 부서별 평균 급여, 최소 급여, 최대 급여를 포함하는 뷰인 dept\_view를 생성하시오.

```
CREATE VIEW dept_view(d_id, d_name, avgsal, minsal, maxsal)
AS SELECT d.dept_id, d.dept_name, AVG(e.sal), MIN(e.sal), MAX(e.sal)
FROM emp e, dept d
WHERE e.dept_id = d.dept_id
GROUP BY d.dept_id, d.dept_name;
```

View created.

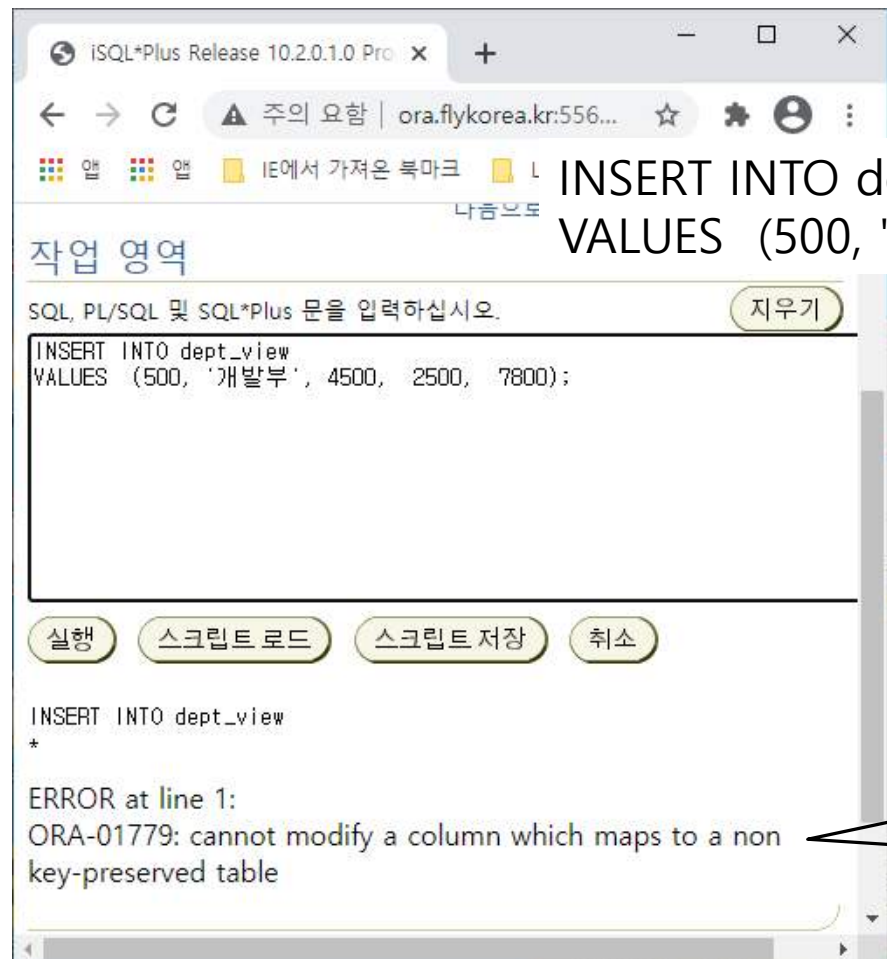
```
SELECT *
FROM dept_view;
```

D_ID	D_NAME	AVGSAL	MINSAL	MAXSAL
100	관리부	4350	2500	6200
300	총무부	3550	2500	5850
200	연구부	5640	2500	9000
400	영업부	4680	3500	7000

## 2. 뷰 생성

### ✓ 주의 : 그룹 함수 값을 포함하는 뷰에서의 삽입 오류

- 그룹 함수 값은 원래 테이블의 데이터가 가공된 것이므로, 이런 경우 뷰를 통한 삽입이 허용되지 않음



The screenshot shows the iSQL\*Plus interface. The SQL editor contains the following text:

```
INSERT INTO dept_view  
VALUES (500, '개발부', 4500, 2500, 7800);
```

Below the editor, the execution buttons are visible: 실행, 스크립트 로드, 스크립트 저장, 취소. The output area shows the following error message:

```
INSERT INTO dept_view  
*  
ERROR at line 1:  
ORA-01779: cannot modify a column which maps to a non  
key-preserved table
```

A callout bubble points to the error message with the text: 뷰에 대한 데이터 조작이 부적합해서 발생한 오류 (500번 부서가 dept 테이블에 없음)

### 3. 뷰 수정

#### ★ 뷰 수정 방법

- 뷰를 생성할 때 사용한 명령인 CREATE OR REPLACE VIEW 명령을 사용해서, 이미 존재하는 뷰를 대체함으로써 뷰를 수정하게 됨

```
CREATE OR REPLACE [FORCE | NOFORCE]
                VIEW 뷰이름 [(열별칭1[, 열별칭2, ...])]
AS 서브쿼리
[WITH CHECK OPTION [CONSTRAINT 제약이름]]
[WITH READ ONLY];
```

#### ★ 뷰 수정의 특징

- 이미 생성된 뷰를 그대로 두고 수정하는 것이 아니라, 이미 생성된 뷰를 제거하고 새로운 뷰를 생성해서 대체함으로써 수정하는 효과를 얻게 됨
- 뷰가 존재하지 않는 경우에도 오류가 발생하지 않고 뷰를 새로 생성함

### 3. 뷰 수정

#### ✓ 뷰 수정 예제

- empview200 뷰의 열 이름을 각각 사번, 이름, 입사로 수정하기

```
CREATE OR REPLACE VIEW empview200  
    (사번, 이름, 입사)
```

```
AS SELECT emp_id, emp_name, hiredate  
    FROM emp  
    WHERE dept_id = 200;
```

View created.

```
SELECT *  
FROM empview200;
```

사번	이름	입사
6200	박철수	17/12/17
6361	김철수	20/06/09
7878	백기수	20/12/09
7933	김철수	21/01/23
7631	박은미	21/03/01

## 4. 뷰 삭제

### ★ 뷰 삭제 방법

- DROP VIEW 명령으로 뷰를 삭제함

```
DROP VIEW view_name;
```

### ★ 뷰 삭제의 특징

- 뷰가 기초하는 기본 테이블에는 영향을 주지않고 뷰만 삭제함
- 즉, 데이터에 전혀 손실을 주지않고, 논리적인 테이블인 뷰를 삭제함
- 삭제된 뷰를 기반으로 생성된 뷰나 어플리케이션은 무효화됨
- 뷰의 생성자 또는 DROP ANY VIEW 권한을 가진 사용자만 삭제 가능함

## 4. 뷰 삭제

### ✓ 뷰 삭제 예제

- empview200 뷰를 삭제한 다음, 뷰를 삭제해도 기본 테이블의 데이터는 보존되는지 확인하시오.

```
DROP VIEW empview200;
```

View dropped.

```
SELECT emp_id, emp_name, hiredate  
FROM emp  
WHERE dept_id = 200;
```

EMP_ID	EMP_NAME	HIREDATE
6200	박철수	17/12/17
6361	김철수	20/06/09
7878	백기수	20/12/09
7933	김철수	21/01/23
7631	박은미	21/03/01



## 5. 뷰 활용

### ★ 뷰 확인 방법

- 생성된 뷰 이름과 뷰 정의는 데이터 사전의 USER\_VIEWS 테이블에 저장됨

```
SELECT view_name  
FROM user_views;
```

VIEW_NAME
EMPVIEW300
SALVIEW100
SALVIEW400
DEPT_VIEW

## 5. 뷰 활용

### ★ 뷰를 통한 데이터 검색

- 뷰를 이용해서 원래 테이블에 있는 데이터를 검색할 수 있음

✓ 예 : 300부서의 사원의 사번과 이름, 업무, 근무지 검색하기

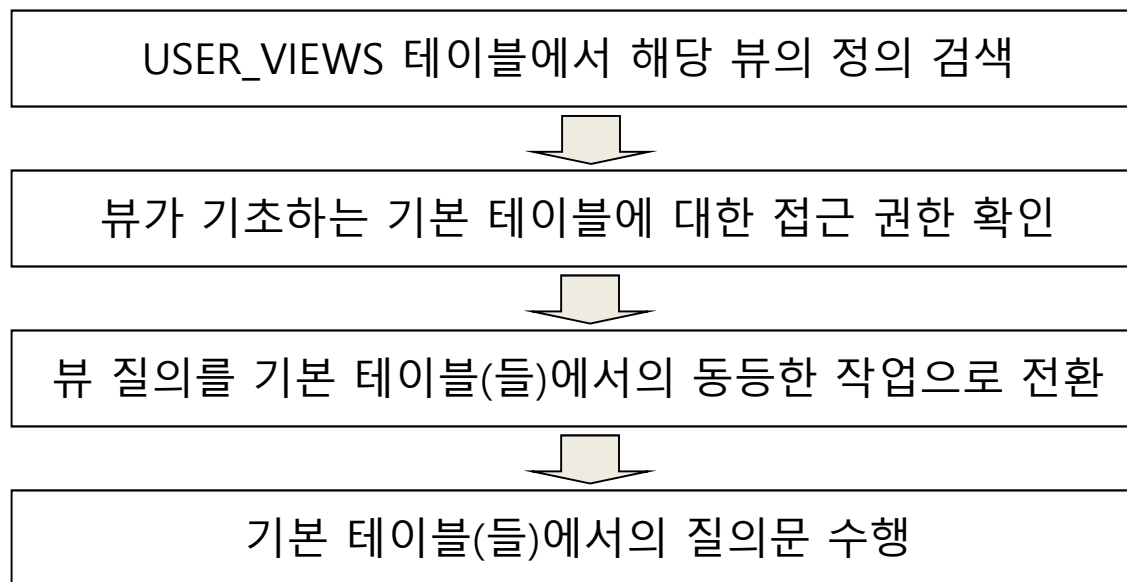
```
SELECT *  
FROM empview300;
```

EMP_ID	EMP_NAME	JOB	DEPT_LOC
6351	진대준	부장	서울
7910	이영희	경리	서울
7920	김마리아	사무직	서울
7696	이혜성	사무직	서울

## 5. 뷰 활용

### ★ 뷰 질의 수행 과정

- 사용자가 뷰를 통해서 데이터에 접근하면, 오라클 서버는 다음과 같은 순서로 처리함으로써 모든 뷰 질의가 기본 테이블에서 수행되도록 처리함



## 5. 뷰 활용

### ★ 뷰에서의 DML 연산 수행 규칙

- 단순 뷰나 복합 뷰를 통한 모든 검색은 허용되지만, 기타 DML 연산의 경우 아래 규칙에 따라 수행됨

① 뷰를 생성할 때 아래 내용이 포함되면 행을 삽입, 삭제, 수정할 수 없음

. 그룹 함수

. GROUP BY 절

. DISTINCT 키워드

-> 원래 테이블의 데이터가 가공된 것이므로

② 뷰가 아래 내용을 포함하면 행을 삽입하거나 수정할 수 없음

. 표현식(예:  $sal * 12$ )으로 정의된 열

-> 표현식의 값이 원래 테이블에는 없으므로

③ 아래 경우에는 뷰에 행을 삽입할 수 없음

. 뷰에 의해 선택되지 않은 NOT NULL 열이 기본 테이블에 있을 때

-> 뷰에서 값을 넣지 않으면 그 열이 널 값을 갖게 되므로

## 5. 뷰 활용

### ✓ 가공된 값을 포함하는 뷰에서의 삭제 예제

- 부서이름과 부서별 평균 급여, 최소 급여, 최대 급여를 포함하는 뷰인 dept\_view에서 100번 부서의 정보를 삭제하시오.

```
SELECT *  
FROM dept_view;
```

D_ID	D_NAME	AVGSAL	MINSAL	MAXSAL
100	관리부	4350	2500	6200
300	총무부	3550	2500	5850
200	연구부	5640	2500	9000
400	영업부	4680	3500	7000

```
DELETE FROM dept_view  
WHERE d_id = 100;
```

```
DELETE FROM dept_view  
*  
  
ERROR at line 1:  
ORA-01732: data manipulation operation not legal on this view
```

뷰에 대한 데이터 조작이 부적합해서  
발생한 오류



**Q & A**