뷰(view)

배재대학교 컴퓨터공학과
김창수

목차

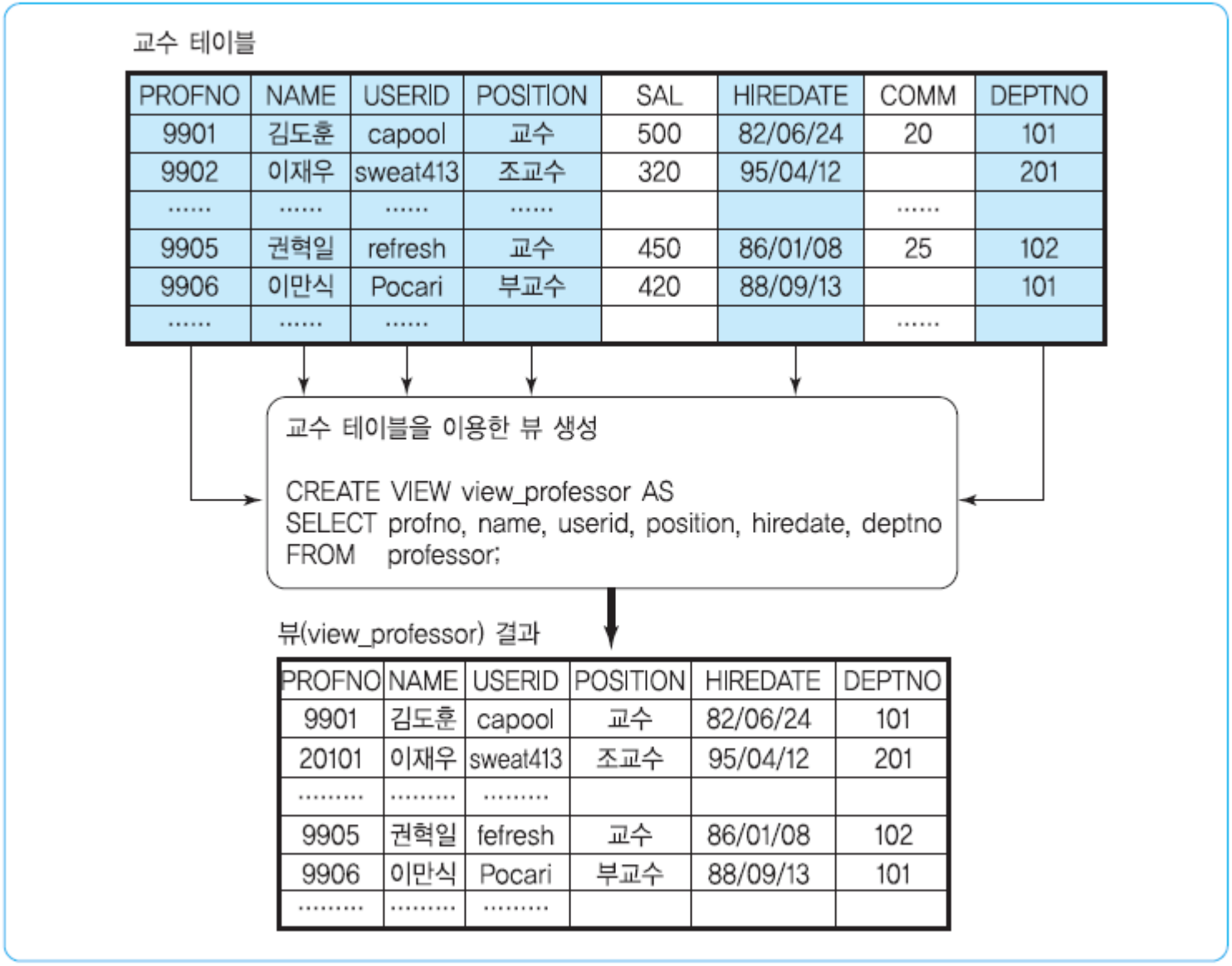
뷰의 개념

- 뷰(view)란?
  - 하나 이상의 기본 테이블이나 다른 뷰를 이용하여 생성되는 가상 테이블
  - 전체의 데이터 중, 일부만 접근할 수 있도록 제한하기 위한 기법
  - 가상 테이블(virtual table)
    - 테이블은 디스크에 공간이 할당되어 데이터를 저장
    - 뷰는 데이터딕셔너리 테이블에 뷰에 대한 정의만 저장
    - 디스크 저장공간이 할당되지 않음
  - 테이블에서 파생된 객체 테이블과 유사하게 사용
    - 오라클에서 뷰를 구성하는 칼럼의 수는 최대 254개까지 허용
  - 뷰에 대한 수정 결과는 뷰를 정의한 기본 테이블에 적용
    - 기본 테이블의 데이터가 변경되면 뷰에도 반영
  - 뷰를 정의한 기본 테이블에서 정의된 무결성 제약 조건 상속
  - 뷰의 정의는 USER_VIEWS 데이터 딕셔너리 테이블에서 조회가능

뷰의 개념

- **뷰의 장점**
  - 데이터를 보호하기 위한 보안(security)
    - 전체 데이터의 일부만 접근할 수 있는 뷰를 정의하여 일반 사용자에게 해당 뷰만 접근 가능하도록 허용하여 중요한 데이터의 유출 방지 가능
    - 예 : 교수 테이블의 급여나 보직수당은 개인적인 정보이므로 학생들의 접근을 제한
  - 사용자 편의성(flexibility)
    - 뷰를 통해 사용자에게 필요한 정보만 선택적으로 제공하여 정보 접근의 편의성 제공 가능
    - 예 : 교수 테이블에서 급여와 보직수당을 학생 입장에서는 불필요한 정보이므로 이를 제외한 칼럼으로 구성된 뷰를 만들어 학생들에게 제공

뷰 생성 개념도


교수 테이블
PROFNO NAME USERID POSITION SAL HIREDATE COMM DEPTNO
9901 김도훈 capool 교수 500 82/06/24 20 101
9902 이재우 sweat413 조교수 320 95/04/12 201
9905 권혁일 refresh 교수 450 86/01/08 25 102
9906 이만식 Pocari 부교수 420 88/09/13 101
교수 테이블을 이용한 뷰 생성
CREATE VIEW view_professor AS
SELECT profno, name, userid, position, hiredate, deptno
FROM professor;
뷰(view_professor) 결과
PROFNO NAME USERID POSITION HIREDATE DEPTNO
9901 김도훈 capool 교수 82/06/24 101
20101 이재우 sweat413 조교수 95/04/12 201
9905 권혁일 fefresh 교수 86/01/08 102
9906 이만식 Pocari 부교수 88/09/13 101

뷰의 종류

- 단순 뷰(simple view)
  - 하나의 기본 테이블에 의해 정의한 뷰
  - 단순 뷰에 DML명령문 실행 가능
  - DML 명령문의 처리 결과는 기본 테이블에 반영됨

- 복합 뷰(complex view)
  - 두개 이상의 기본 테이블로 구성한 뷰
  - 무결성 제약조건, 표현식, GROUP BY 절의 유무에 따라 DML명령문의 제한적으로 사용 가능
  - 복합 뷰는 DISTINCT, 그룹 함수, GROUP BY, START WITH CONNECT BY, ROWNUM 을 포함할 수 없음
  - UNION ALL, INTERSECT 등과 같은 집합 연산 실행 불가

목차

뷰 생성

- **뷰 생성**
  - CREATE VIEW **명령문 사용**
  - **뷰 생성시 칼럼 이름을 명시하지 않으면 기본 테이블의 칼럼 이름을 상속**
  - **함수나 표현식에 의해 정의된 칼럼은 별도의 이름을 반드시 명시해야 함**
    - **별도의 이름을 사용하지 않으면 에러가 발생하므로 별명(alias) 사용 필요**

사용법

```
CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW view
[(alias, alias, ...)]
AS subquery
```

- OR REPLACE : 기존 뷰와 동일한 이름으로 뷰를 재생성하는 경우
- FORCE : 기본 테이블의 존재 여부에 상관없이 뷰 생성
- NOFORCE : 기본 테이블이 존재할 경우에만 뷰 생성, 기본 값
- ALIAS : 기본 테이블의 칼럼 이름과 다르게 지정한 뷰의 칼럼 이름

단순 뷰 생성

학생 테이블에서 101번 학과 학생들의 학번, 이름, 학과번호로 정의되는 단순 뷰를 생성하여라.

```
SQL> CREATE VIEW v_stud_dept101(학번, 이름, 학과번호)
  2  AS SELECT studno, name, deptno
  3     FROM   student
  4     WHERE  deptno = 101;

뷰가 생성되었습니다.

SQL> SELECT * FROM v_stud_dept101;
```

뷰 생성시 칼럼 이름을 명시하지 않으면 칼럼 이름은 studno, name, deptno가 된다.

☞ 출력 결과

101번 소속의 학생 데이터만 출력된다.

학번	이름	학과번호
10101	전인하	101
10102	박미경	101
10103	김영균	101
10104	지은경	101
10105	임유진	101
10106	서재진	101
10107	이광훈	101
10108	류민정	101

8 개의 행이 선택되었습니다.

복합 뷰 생성

학생 테이블과 부서 테이블을 조인하여 102번 학과 학생들의 학번, 이름, 학년, 학과 이름으로 정의되는 복합 뷰를 생성하여라.

```
SQL> CREATE VIEW v_stud_dept102(학번, 이름, 학년, 학과이름)
  2  AS SELECT s.studno, s.name, s.grade, d.dname
  3  FROM student s, department d
  4  WHERE s.deptno=d.deptno AND s.deptno=102;

SQL> SELECT * FROM v_stud_dept102;
```

102번 학과 학생에 대한 조인 결과만 출력된다.

☞ 출력 결과

```
      학번 이름       학년 학과이름
---------- ---------- ---- ----------------
     10202 오유석       4  멀티미디어학과
     10201 김진영       2  멀티미디어학과
     10203 하나리       1  멀티미디어학과
     10204 윤진욱       3  멀티미디어학과
```

함수를 사용한 뷰 생성 사용 예

사용예

교수 테이블에서 학과별 평균 급여와 총계로 정의되는 뷰를 생성하여라.

```
SQL> CREATE VIEW v_prof_avg_sal
  2  AS SELECT deptno, SUM(sal), AVG(sal)
  3        FROM professor
  4        GROUP BY deptno;

AS SELECT deptno, SUM(sal), AVG(sal)
                  *
2행에 오류:
ORA-00998: 이 식은 열의 별명과 함께 지정해야 합니다

SQL> CREATE VIEW v_prof_avg_sal
  2  AS SELECT deptno, SUM(sal) sum_sal, AVG(sal) avg_sal
  3        FROM professor
  4        GROUP BY deptno;

뷰가 생성되었습니다.
```

함수를 사용하여 뷰를 생성하는 경우, 칼럼 별명을 사용하지 않으면 오류가 발생한다.

인라인 뷰

- **인라인 뷰**(inline view)
  - FROM 절에서 서브쿼리를 사용하여 생성한 임시 뷰
  - SQL 명령문이 실행되는 동안만 임시적으로 정의
  - FROM 절에서 참조하는 테이블의 크기가 클 경우, 필요한 행과 칼럼만으로 구성된 집합을 재정의하여 질의문을 효율적 구성

FROM 절에서 서브쿼리 사용

```
SELECT column_list
FROM   (subquery) alias
WHERE  condition;
```

인라인 뷰 사용 예1

사용예

인라인 뷰를 사용하여 학과별로 학생들의 평균 키와 평균 몸무게, 학과 이름을 출력하여라.

```
SQL> SELECT dname, avg_height, avg_weight
  2  FROM (SELECT deptno, avg(height) avg_height,
  3                avg(weight) avg_weight
  4        FROM   student
  5        GROUP BY deptno) s, department d
  6  WHERE s.deptno = d.deptno ;
```

☞ 출력 결과

```
DNAME          AVG_HEIGHT AVG_WEIGHT
-------------- ---------- ----------
컴퓨터공학과       171.125         68
멀티미디어학과         168       69.5
전자공학과             176      61.75
```

인라인 뷰 사용 예2

사용예

인라인 뷰를 사용하여 모든 학생 중에서 몸무게가 적은순으로 상위 5명을 출력하여라. RANK( ) 함수는 출력 결과에 순위를 부여하는 함수로 16장에서 설명한다.

```
SQL> SELECT studno, name, weight, rn
 2   FROM (SELECT studno, name, weight,
 3                RANK() OVER (ORDER BY weight) rn
 4         FROM student)
 5   WHERE rn BETWEEN 1 AND 5;
```

☞ 출력 결과

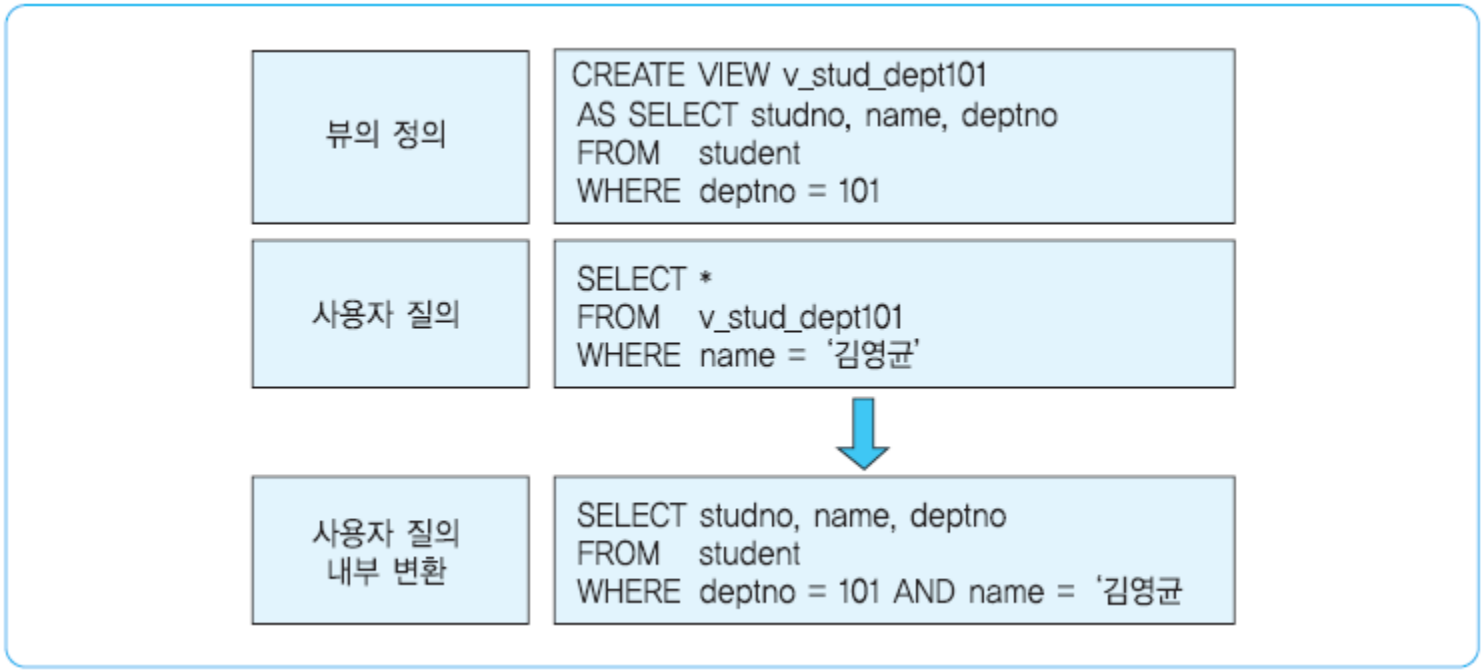
STUDNO	NAME	WEIGHT	RN
10104	지은경	42	1
10201	김진영	48	2
20103	김진경	51	3
10102	박미경	52	4
10105	임유진	54	5

뷰의 내부 처리 과정

- 뷰에 대한 질의의 내부적인 처리 과정
  - 뷰는 디스크상에 저장된 공간이나 데이터가 없는 가상 테이블이므로 실체가 없음
  - 뷰에 대한 질의는 내부적으로 뷰를 정의한 기본 테이블에 대한 질의로 변환되어 실행
  - 뷰에 대한 질의가 내부적으로 처리되는 과정
    - 1) USER_VIEW 데이터 딕셔너리에서 뷰에 대한 정의 조회
    - 2) 기본 테이블에 대한 뷰의 접근 권한 확인
    - 3) 뷰에 대한 질의를 기본 테이블에 대한 질의로 변환
    - 4) 기본 테이블에 대한 질의를 통해 데이터 검색
    - 5) 검색된 결과 출력

뷰의 내부 처리 과정

- 뷰에 대한 질의의 내부적인 처리 과정



뷰와 관련된 데이터 딕셔너리 테이블

- USER_VIEWS
  - 사용자가 생성한 모든 뷰에 대한 정의를 저장

```
SQL> SELECT view_name, text
  2  FROM   user_views;
```

USER_VIEWS 데이터 딕셔너리에 뷰를 정의한 SELECT 명령문이 문자열로 저장된다.

☞ 출력 결과

```
VIEW_NAME            TEXT
-------------------- ------------------------------------------------
V_PROF_AVG_SAL       SELECT deptno, SUM(sal) sum_sal, AVG(sal) avg_sal
                     FROM professor
                     GROUP BY deptno

V_STUD_DEPT101       SELECT studno, name, deptno
                     FROM student
                     WHERE deptno = 101
V_STUD_DEPT102       SELECT s.studno, s.name, s.grade, d.dname
                     FROM student s, department d
                     WHERE s.deptno = d.deptno AND s.deptno = 102
```

뷰의 변경

- **뷰의 변경**
  - 뷰에 대한 정의를 수정하는 것
  - 기존 뷰에 대한 정의를 삭제한 후 재생성 하거나 CREATE 명령문에서 “OR REPLACE” 옵션을 이용하여 재정의 가능
    - OR REPLACE 옵션은 기존에 생성된 뷰가 있을 경우, 기존 뷰를 무시하고 재생성하는 옵션
  - 변경 결과는 USER_VIEWS 데이터 딕셔너리에 저장

뷰의 변경 사용 예

사용예

기존 V_STUD_DEPT101 뷰에 학년 칼럼을 추가하여 재정의하여라.

```
SQL> CREATE OR REPLACE VIEW v_stud_dept101
  2  (학번, 이름, 학과번호, 학년)
  3  AS SELECT studno, name, deptno, grade
  4     FROM   student
  5     WHERE  deptno = 101;

뷰가 생성되었습니다.
```

뷰 정의 확인

```
SQL> DESC v_stud_dept101
```

☞ 출력 결과

이름	널?	유형
학번	NOT NULL	NUMBER(5)
이름	NOT NULL	VARCHAR2(10)
학과번호		NUMBER(4)
학년		VARCHAR2(1)

뷰에 대한 데이터 조작

- **단순 뷰**
  - 단순 뷰는 기본 테이블과 동일하게 DML 명령문 사용 가능
    - 뷰에 대한 DML명령문은 내부적으로는 기본 테이블에 대한 데이터를 조작하는 과정
    - 뷰에 대한 무결성 제약조건도 기본 테이블에 정의된 무결성 제약조건이 적용
- **복합 뷰**
  - 복합 뷰에서는 일부 DML 명령어의 사용 제한
- **데이터 조작이 불가능한 경우**
  - 뷰 정의에 포함되지 않는 기본 테이블의 칼럼이 NOT NULL 제약조건으로 지정된 경우 데이터 삽입 불가
  - 뷰 정의시 표현식으로 정의된 칼럼에 대해서는 UPDATE, INSERT 명령문의 실행이 불가능
  - 뷰 정의시 그룹 함수, DISTINCT, GROUP BY 절을 포함한 경우에는 모든 종류의 DML 명령문 사용 불가

뷰의 삭제

- **뷰의 삭제**
  - 뷰의 삭제는 USER_VIEWS데이터 딕셔너리에 저장된 뷰의 정의를 삭제하는 것
  - 뷰를 정의한 기본 테이블의 구조나 데이터에는 전혀 영향 없음

사용법

```
DROP VIEW view;
```

뷰의 삭제 사용 예

사용예

학생 테이블과 연관된 모든 뷰를 삭제하여라.

```
SQL> DROP VIEW v_stud_dept101;
뷰가 삭제되었습니다.

SQL> DROP VIEW v_stud_dept102;
뷰가 삭제되었습니다.

SQL> DROP VIEW v_prof_avg_sal;
뷰가 삭제되었습니다.

SQL> SELECT view_name, text
  2  FROM user_views;
선택된 레코드가 없습니다.
```