11강 뷰

뷰

- 일반적으로 데이터를 저장하기 위해 테이블을 사용한다.
- 테이블에 저장된 데이터는 실제 디스크 공간에 할당되어서 저장이 된다.
- 그러나 테이블과 다르게 뷰는 테이블과 비슷한 형태를 가지면서도 디스크에는 실제 저장하지 않는 '가상의 테이블'이다.

- 뷰를 사용하는 이유
  - 사용자 권한 제어 => 보안에 도움
  - 쿼리를 보다 단순하게 사용가능

뷰

- 뷰를 정의 하기 위해 사용하는 테이블을 기본 테이블이라고 부른다.
- 뷰는 기본적으로 읽기전용으로 사용하지만 데이터 수정도 가능하다.
- 데이터 수정시 그 결과는 기본 테이블에 반영이 된다.
- 반대로 기본 테이블에 데이터가 변경되어도 뷰에 반영이 된다.

뷰

- 뷰를 생성해 보자
- 뷰의 사용법
  - CREATE OR REPLACE VIEW 뷰이름 (컬럼명1, 컬럼명2....)
  AS subquery;

  - 실습을 위한 테이블을 준비한다.(이미 존재하는 테이블있는 경우 삭제 한다.)

```
CREATE TABLE emp_second
AS
SELECT * FROM employee;

CREATE TABLE dept_second
AS
SELECT * FROM department;
```

뷰

- 실습에 앞서 뷰의 종류부터 살펴 보면
  - 단순 뷰 : 하나의 기본 테이블로 생성된 뷰 – DML명령을 수행 할 수 있다.
  - 복합 뷰 : 두개 이상의 기본테이블로 생성된 뷰 – 여러가지 제약조건, 표현식, 그룹화에 의해 DML 사용이 제한적이다.
    - 복합 뷰는 DISTINCT나 그룹함수, GROUP BY 절 등을 사용할 수 없다.

뷰

- 단순 뷰를 작성해보자
  - 담당 업무가 SALESMAN인 사원들의 레코드를 담는 뷰를 만들어 본다 컬럼은 사원번호, 사원이름, 부서번호, 담당업무로 제한한다.

```
CREATE OR REPLACE VIEW v_emp_job(사번, 사원이름, 부서번호, 담당업무)
AS
SELECT eno, ename, dno, job
FROM emp_second
WHERE job like 'SALESMAN';
```

- 생성한 뷰를 사용해 보자

```
SELECT *
FROM v_emp_job;
```

뷰

- 뷰를 생성할 때 칼럼명을 생략하면 기본 칼럼명을 사용한다.

```
CREATE OR REPLACE VIEW v_emp_job2
AS
SELECT eno, ename, dno, job
FROM emp_second
WHERE job like 'SALESMAN';
```

- 생성한 뷰를 사용해 보자

```
SELECT *
FROM v_emp_job2;
```

뷰

- 복합 뷰를 생성해보자
  - dept_second테이블과 emp_second테이블을 natural 조인을 한 결과를 뷰에 담아 본다

```
CREATE OR REPLACE VIEW v_emp_complex
AS
SELECT *
FROM emp_second NATURAL JOIN dept_second;
```

- 생성한 뷰를 사용해 보자

```
SELECT *
FROM v_emp_complex;
```

뷰 - 필요성

- 보안을 위한 뷰를 생성해 보자
  - 일반 사용자에게 뷰에만 접근하도록 해서 기본테이블에 저장된 민감한 데이터에 접근을 막는다(예 : 급여정보등..)

```
CREATE OR REPLACE VIEW v_emp_sample
AS
SELECT eno, ename,manager ,job, dno
FROM emp_second;
```

- 정보를 쉽게 사용하기 위한 뷰를 생성해보자
  - 매번 조인을 통해서 데이터를 얻는다면 매번 복잡한 쿼리를 작성해야 하는데 이런 복잡한 쿼리를 미리 뷰로 생성해 두면 뷰만 호출함으로 정보를 쉽게 사용할 수 있다.

```
CREATE OR REPLACE VIEW v_emp_complex2
AS
SELECT e.eno, e.ename, e.salary, dno, d.dname, d.loc
FROM emp_second e NATURAL JOIN dept_second d;
```

뷰의 정체

- 뷰는 데이터를 가지고 있지 않는 가상의 테이블이므로 실체가 없다.
- 단순히 AS 절 이하 쿼리문을 저장하고 있다가 매번 쿼리문을 실행함으로 데이터를 불러온다.
  - 뷰 데이터 사전 살펴보기

```
SELECT view_name, text
FROM user_views;
```

  - 뷰가 처리되는 과저

뷰의 처리 과정

- 뷰가 처리되는 과정
  - USER_VIEW 데이터 사전에서 뷰에 대한 정의 조회
  - 기본테이블에 대한 뷰의 접근 권한을 살핌
  - 뷰에 대한 질의를 기본 테이블에 대한 질의로 변환
  - 기본 테이블에 대한 질의를 통해 데이터 검색
  - 검색 결과 출력

- 뷰는 기본테이블을 간접적으로 접근하므로 기본적인 조회 기능 뿐 아니라 데이터 삽입, 변경, 삭제를 할 수 있다.

```
INSERT INTO v_emp_job
VALUES(9000,'Ahn',30,'SALESMAN');
```

```
SELECT * FROM v_emp_job;
```

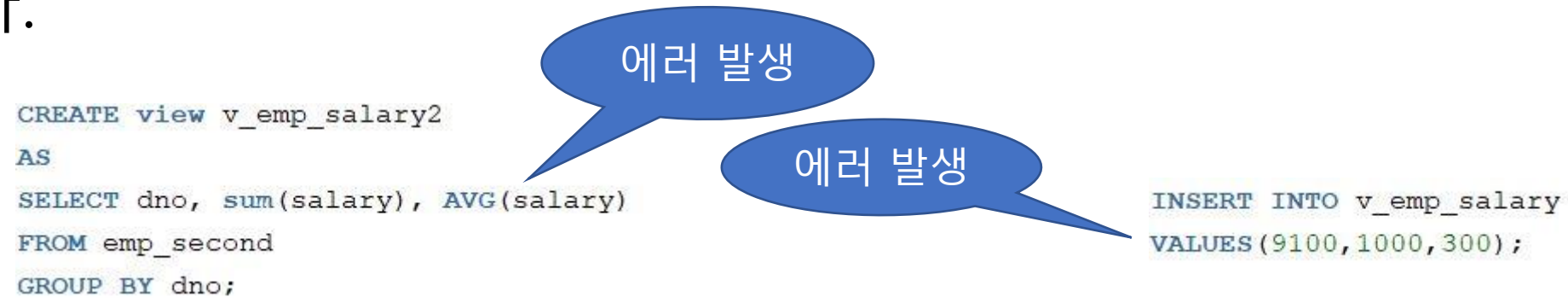
```
SELECT *
FROM emp_second
ORDER BY eno ASC;
```

다양한 뷰 생성

- 함수를 사용하여 뷰를 생성할 수도 있다
- 그러나 그룹함수를 사용하는 경우 물리적인 칼럼이 존재하지 않고 가상의 칼럼을 사용하기 때문에 반드시 별칭을 사용해야 한다.

```
CREATE view v_emp_salary
AS
SELECT dno, sum(salary) AS "sal_sum", AVG(salary) AS "sal_avg"
FROM emp_second
GROUP BY dno;
SELECT * FROM v_emp_salary;
```

- 단 그룹함수를 가상 컬럼으로 가지는 뷰는 DML을 사용할 수 없다.



뷰 제거하기

- 뷰가 더 이상 필요 없을 때는 DROP 명령으로 제거 가능

```
DROP VIEW v_emp_job;

SELECT view_name, text
FROM user_views;
```

뷰의 옵션 – OR REPLACE

- OR REPLACE : 옵션은 뷰가 이미 존재한다면 뷰의 내용을 갱신하기 위한 옵션


```
CREATE VIEW v_emp_job2
AS
SELECT eno, ename, dno, job
FROM emp_second
WHERE job like 'MANAGER';
```

```
CREATE OR REPLACE VIEW v_emp_job2
AS
SELECT eno, ename, dno, job
FROM emp_second
WHERE job like 'MANAGER';
```

뷰의 옵션 – FORCE

- FORCE : 옵션은 기본 테이블의 존재 유무와 상관없이 뷰를 생성할 때 사용한다.
- 기본값은 NOFORCE => 뷰 생성시 반드시 기본 테이블이 있어야 한다.

```
CREATE OR REPLACE VIEW v_emp_notable
AS
SELECT eno, ename, dno, job
FROM emp_notable
WHERE job like 'MANAGER';
```

에러 발생

```
CREATE OR REPLACE FORCE VIEW v_emp_notable
AS
SELECT eno, ename, dno, job
FROM emp_notable
WHERE job like 'MANAGER';
```

뷰의 옵션 – FORCE

- WITH CHECK OPTION : 해당 뷰를 통해서 볼 수 있는 범위 내에서만 UPDATE나 INSERT가 가능하다.

```
CREATE OR REPLACE VIEW v_emp_job_nochk
AS
SELECT eno, ename, dno, job
FROM emp_second
WHERE job like 'MANAGER';

INSERT INTO v_emp_job_nochk
VALUES(9200,'LEE',30,'SALESMAN');

SELECT * FROM v_emp_job_nochk;
```

삽입 가능

조회 불가능

뷰의 옵션 – WITH CHECK OPTION

- WITH CHECK OPTION : 해당 뷰를 통해서 볼 수 있는 범위 내에서만 UPDATE나 INSERT가 가능하다.

```
CREATE OR REPLACE VIEW v_emp_job_chk
AS
SELECT eno, ename, dno, job
FROM emp_second
WHERE job like 'MANAGER' WITH CHECK OPTION;

INSERT INTO v_emp_job_chk
VALUES(9300,'PARK',30,'SALESMAN');

INSERT INTO v_emp_job_chk
VALUES(9300,'PARK',30,'MANAGER');

SELECT * FROM v_emp_job_chk;
```

오류 발생

오류 해결

뷰의 옵션 – WITH READ ONLY

- WITH READ ONLY : 해당 뷰를 통해서 SELECT만 가능하고 UPDATE나 INSERT가 불가능 하게 해주는 옵션이다.

```
CREATE OR REPLACE VIEW v_emp_job_readonly
AS
SELECT eno, ename, dno, job
FROM emp_second
WHERE job like 'MANAGER' WITH READ ONLY;


INSERT INTO v_emp_job_readonly
VALUES(9400,'LIM',30,'MANAGER');
```