가상 테이블인 뷰

뷰의 기본 다루기

❖ **뷰**

- 물리적인 테이블에 근거한 논리적인 가상 테이블
- 실질적으로 데이터를 저장하고 있지 않음
- 테이블을 사용하는 것과 동일하게 뷰 사용 가능

뷰의 기본 다루기

❖ 뷰 테이블 만들기

ㅇ 형식

```
CREATE [OR REPLACE] [{FORCE|NOFORCE}] VIEW 뷰이름
       [(alias, alias, alias, …)]
       AS 서브쿼리
       [WITH CHECK OPTION]
       [WITH READ ONLY];
```

- CREATE VIEW : 뷰의 구조 변경시 삭제후 다시 생성
- CREATE OR REPLACE VIEW : 기존 뷰를 새로운 구조의 뷰로 변경 가능
- FORCE | NOFORCE : FORCE를 사용하면 기본 테이블의 존재 여부와 상관없이 뷰 생성
- WITH CHECK OPTION : 해당 뷰를 통해서 볼 수 있는 범위 내에서만 UPDATE 또는 INSERT가 가능
- WITH READ ONLY: 해당 뷰를 통해서는 SELECT만 가능,
  생략시 추가,수정, 삭제(INSERT, UPDATE, DELETE)가 모두 가능

뷰의 기본 다루기

❖ 뷰정의하기

- ㅇ 테스트 테이블 준비

```
CREATE TABLE DEPT_COPY
AS
SELECT * FROM DEPARTMENTS;

CREATE TABLE EMP_COPY
AS
SELECT * FROM EMPLOYEES;
```

뷰의 기본 다루기

❖ 뷰의 필요성

- 30번 부소에 소속된 사원들의 사번, 이름, 부서 번호를 자주 검색하는 경우

```
 SELECT employee_id, first_name, last_name, department_id
FROM emp_copy
WHERE department_id = 30;
```

- SELECT 문을 하나의 뷰로 정의하여 뷰 조회

```
CREATE VIEW EMP_VIEW30
AS
SELECT employee_id, first_name, last_name, department_id
FROM emp_copy
WHERE department_id = 30;

SELECT * FROM EMP_VIEW30;
```

뷰의 기본 다루기

❖ 뷰 생성 권한이 없는 경우

- SYS 계정으로 권한 부여

```
conn sys/####

GRANT CREATE VIEW TO hr;
```

❖ 실습

- 기본 테이블 EMP_COPY에서 20번 부서에 속한 사원들의 사번, 이름, 부서 번호, 상관의 사번을 출력하기 위한 EMP_VIEW20이라는 뷰를 정의하라.

뷰의 고급 다루기

❖ 컬럼에 별칭 부여하기

```
CREATE OR REPLACE
VIEW EMP_VIEW(사원번호, 사원명, 급여, 부서번호)
AS
SELECT employee_id, first_name || ' ' || last_name, salary,
        department_id
FROM emp_copy;

SELECT * FROM EMP_VIEW;
```

❖ 뷰의 확인

```
SELECT VIEW_NAME, TEXT
FROM USER_VIEWS;
```

뷰의 고급 다루기

❖ 복합 뷰 만들기

- 조인된 테이블을 기본 테이블로하여 뷰 만들기

```
CREATE OR REPLACE
VIEW EMP_DEPT_VIEW
AS
SELECT e.employee_id, e.first_name || ' ' || e.last_name full_name,
        e.salary, d.department_id, d.department_name
FROM emp_copy e, dept_copy d
WHERE e.department_id = d.department_id
ORDER BY e.employee_id DESC;

SELECT * FROM EMP_VIEW;
```

뷰의 고급 다루기

❖ 뷰 삭제하기

- 형식
  - DROP VIEW 뷰이름

```
DROP VIEW VIEW_SAL;
```

뷰 활용하여 Top-N 구하기

❖ 뷰를 활용한 TOP-N 구하기

- 사원 중에서 입사일이 빠른 사람 5명(TOP-5)만 얻는 경우
- ROWNUM과 인라인 뷰 사용

❖ 테이블 생성시 자동으로 제공되는 컬럼(의사 컬럼)

- ROWID : 로우를 유일하게 구분해주는 ID (로우의 논리적 주소)
- ROWNUM : 각 행에 대한 일련 번호

```
SELECT rownum, employee_id, last_name, hire_date
FROM employees;

SELECT rownum, employee_id, last_name, hire_date
FROM employees
ORDER BY hire_date;
```

뷰 활용하여 Top-N 구하기

❖ 뷰와 ROWNUM 컬럼으로 TOP-N 구하기

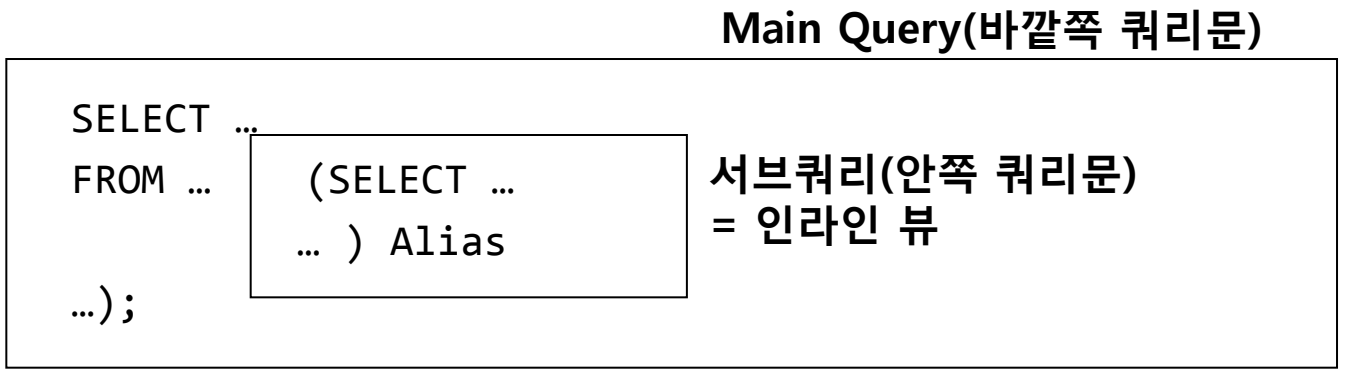
- 입사일을 기준으로 정렬된 뷰 만들기

```
CREATE OR REPLACE VIEW VIEW_HIRE
AS
SELECT employee_id, last_name, hire_date
FROM employees
ORDER BY hire_date;

SELECT rownum, employee_id, last_name, hire_date
FROM view_hire
WHERE rownum <=5 ;
```

뷰 활용하여 Top-N 구하기

❖ 인라인 뷰로 구하는 TOP-N의 개념




```
SELECT rownum, employee_id, last_name, hire_date
FROM (  SELECT employee_id, last_name, hire_date
        FROM employees
        ORDER BY hire_date)
WHERE rownum <= 5;
```

뷰 활용하여 Top-N 구하기

❖ **실습**

- 인라인 뷰를 사용하여 급여를 많이 받는 순서대로 3명만 출력하는 뷰(SAL_TOP_VIEW)를 작성하세요.

```
CREATE OR REPLACE VIEW SAL_TOP5_VIEW
AS
SELECT ROWNUM AS RANKING, employee_id, last_name, salary
FROM (  SELECT employee_id, last_name, salary
        FROM emp_copy
        WHERE salary IS NOT NULL
        ORDER BY salary DESC)
WHERE ROWNUM <=3;
```

```
SELECT *
FROM (
    SELECT  ROW_NUMBER() OVER (ORDER BY USERID ) AS SEQ,
                USERID, UNAME
    FROM MEMBERS
)
WHERE   SEQ  BETWEEN  21 AND 30;
```