11. 뷰(View)

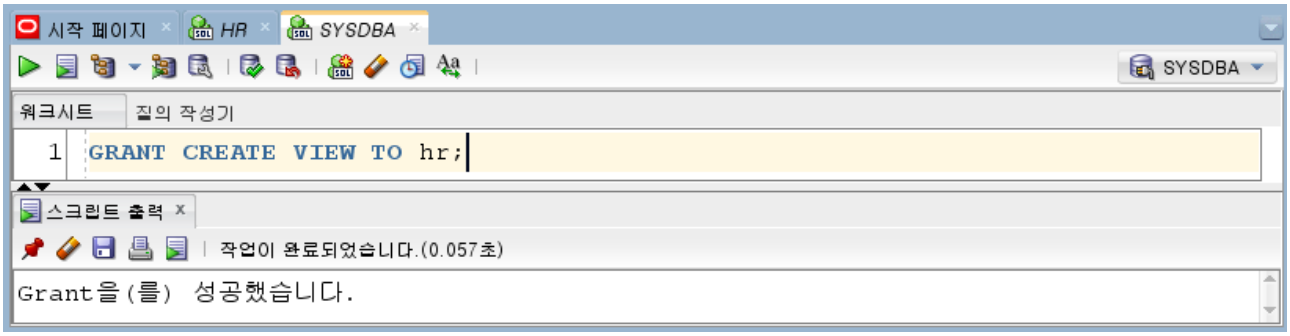
보안과 복잡한 쿼리문을 대신할 뷰를 생성하고 조회한다.
뷰를 제거 변경한다.
인라인 뷰를 이용하여 TOP-N을 구한다.

1) **뷰의 개념**

뷰(View)는 한마디로 물리적인 테이블을 근거한 논리적인 가상 테이블이라고 정의할 수 있다.
뷰는 기본 테이블에서 파생된 객체로서 기본 테이블에 대한 하나의 쿼리문이다.
뷰(View)란 **'보다'란 의미를 갖고 있는 점을 감안해 보면 알 수 있듯이 실제 테이블에 저장된 데이터를 뷰를 통해서 볼 수 있도록 한다.**

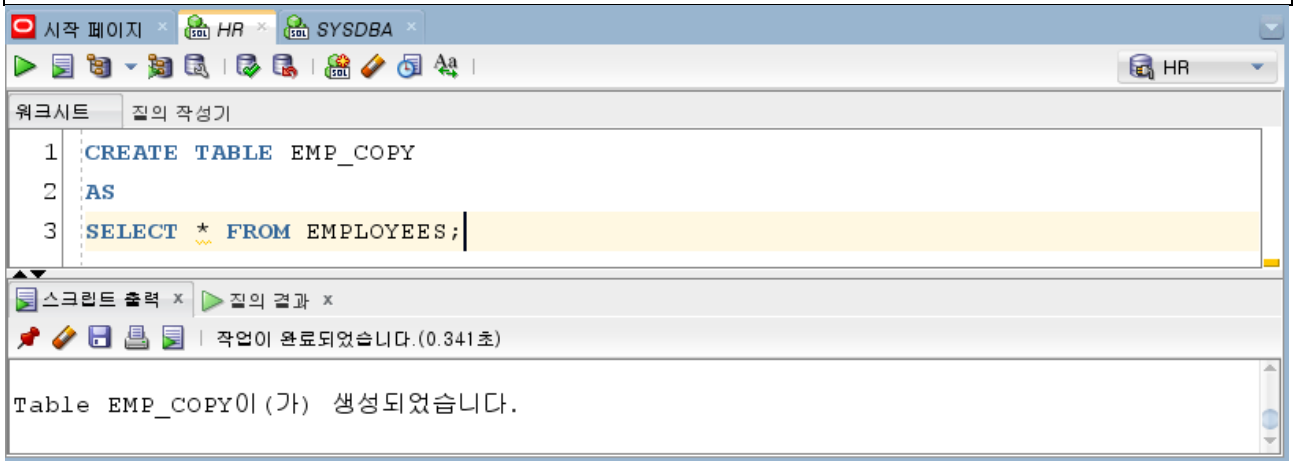
뷰는 물리적인 구조인 테이블과는 달리 데이터 저장 공간이 없다. **뷰는 단지 쿼리문을 저장하고 있는 객체라고 표현할 수 있다.**

```
-- 실행시 오류가 발생하면 권한 문제이기에 SYS로 접근하여 권한을 부여한다.
sys 로 접속한다.
GRANT CREATE VIEW TO hr;
```

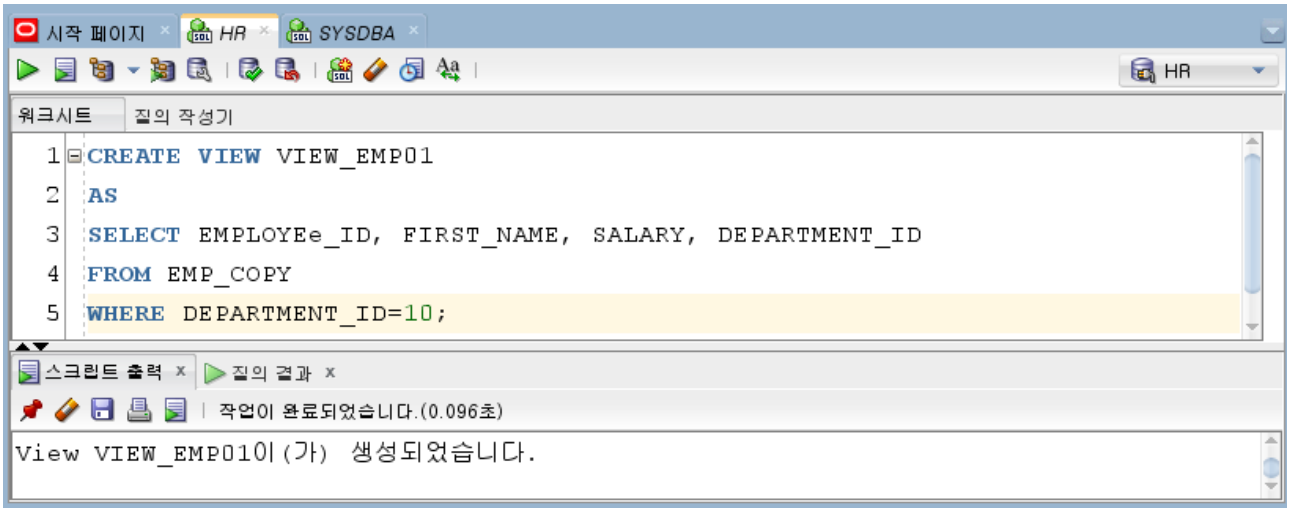


HR로 접속한다.

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



```
CREATE VIEW VIEW_EMP01
AS
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMP_COPY
WHERE DEPARTMENT_ID=10;
```



• 뷰의 사용 목적

직접적인 테이블 접근을 제한하기 위해서 사용된다.

복잡한 질의를 쉽게 만들기 위해서 사용된다.

• 뷰의 특징

뷰는 테이블에 대한 제한을 가지고 테이블의 일정한 부분만 보일 수 있는 가상의 테이블이다.

뷰는 실제 자료를 갖지는 않지만, 뷰를 통해 테이블을 관리할 수 있다. 하나의 테이블에 뷰의 개수는 제한이 없다.

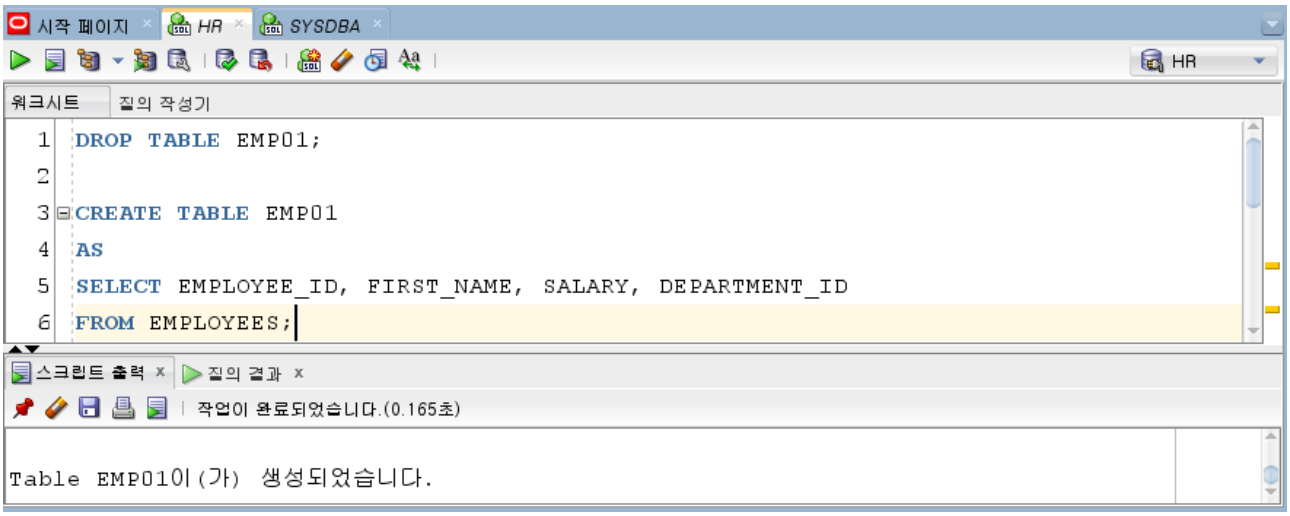
2) 뷰 생성과 조회

뷰를 생성하기 위해서는 테이블 생성과 같이 CREATE문을 사용한다.

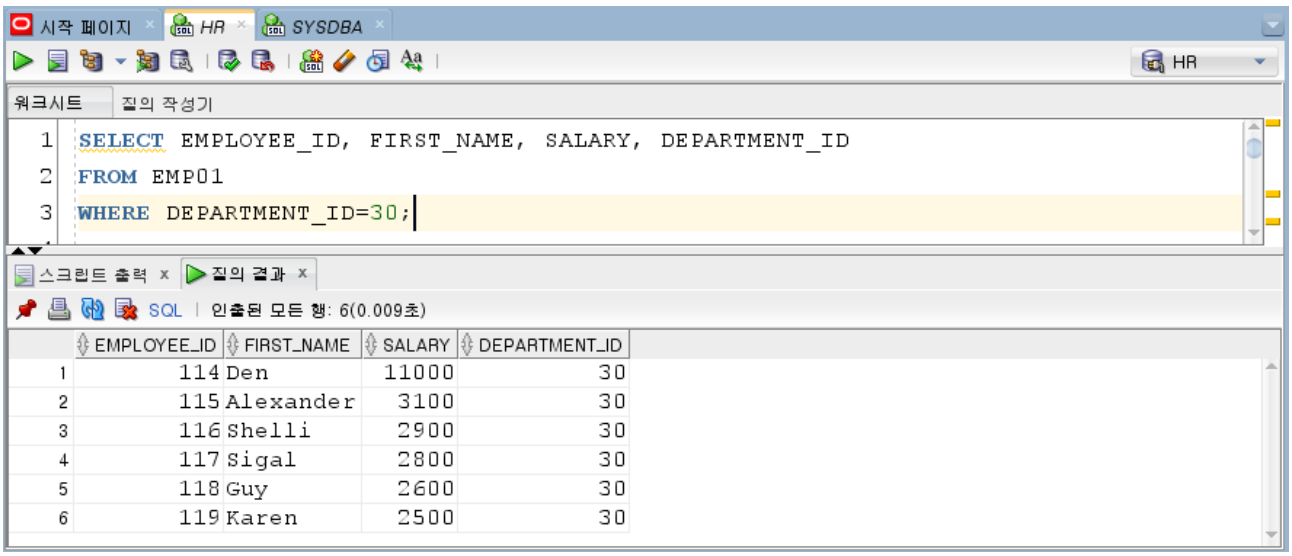
• 기본 테이블

뷰에 의해 제한적으로 접근해서 사용하는 실질적으로 데이터를 저장하고 있는 물리적인 테이블을 말한다.

```
DROP TABLE EMP01;

CREATE TABLE EMP01
AS
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMPLOYEES;
```




```
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMP01
WHERE DEPARTMENT_ID=30;
```



3) 뷰 생성

뷰는 테이블처럼 하나의 개체로서 테이블을 생성할 때와 유사하게 CREATE VIEW 명령어로 생성한다.

```
CREATE [OR REPLACE] [FORCE | NOFORCE] VIEW view_name
[(alias, alias, alias, ...)]
AS subquery
[WITH CHECK OPTION]
[WITH READ ONLY];          -- 해당 뷰를 통해서는 select만 가능하며
                           -- insert/update/delete를 할 수 없다
```

• OR RELPACE VIEW

새로운 뷰를 만들 수 있을 뿐만 아니라 기존에 뷰가 존재하더라도 삭제하지 않고 새로운 구조의 뷰로

변경(REPLACE)할 수 있다.

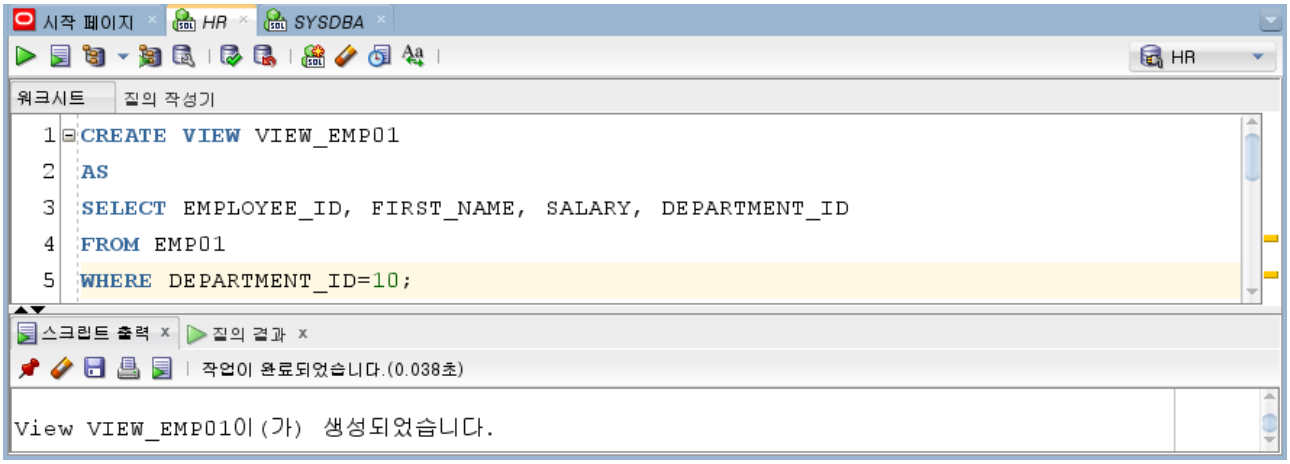
• **FORCE**

기본 테이블의 존재 여부에 상관없이 뷰를 생성한다.

• **WITH CHECK OPTION**

해당 뷰를 통해서 볼 수 있는 범위 내에서만 UPDATE 또는 INSERT가 가능하다.

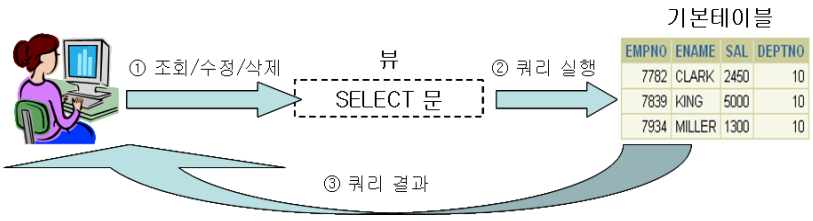
```
CREATE VIEW VIEW_EMP01
AS
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMP01
WHERE DEPARTMENT_ID=10;
```



사용자는 뷰를 그냥 테이블처럼 생각하고 접근(SELECT)하면 된다.

그러면 오라클 서버가 알아서 처리해준다.

뷰는 실질적인 데이터를 저장한 기본 테이블을 볼 수 있는 투명한 창일 뿐이다.



뷰를 생성할 권한이 불충분하다고 오류가 발생하는 경우가 있다.

이럴 경우에는 DBA인 SYS 계정으로 로그인하여 뷰를 생성할 권한을 부여한다.

특정 사용자에 대해서 아무 문제없이 뷰가 생성된다면 괜찮지만 그렇지 않을 경우 GRANT 명령어로 특정 사용자에게 권한을 부여해야 한다. hr 사용자에게 테이블을 생성할 CREATE VIEW 권한을 부여하는 명령어이다.

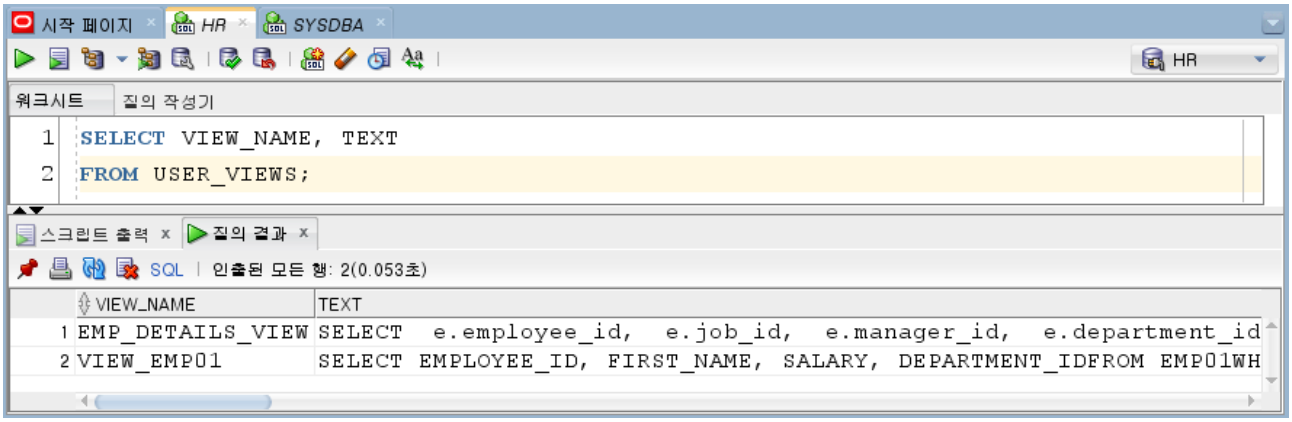
```
GRANT CREATE VIEW TO hr;
```

① 뷰에 관련된 데이터 딕셔너리

데이터 딕셔너리 **USER_VIEWS**에 사용자가 생성한 모든 뷰에 대한 정의가 저장되어 있다.

뷰의 이름을 위한 VIEW_NAME이란 칼럼과 뷰를 작성할 때 기술한 서브 쿼리문이 저장되어 있는 TEXT 칼럼에 대해서 살펴보면

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



② 뷰의 동작 원리

- 사용자가 뷰에 대해서 질의를 하면 USER_VIEWS에서 뷰에 대한 정의를 조회한다.
- 기본 테이블에 대한 뷰의 접근 권한을 살핀다.
- 뷰에 대한 질의를 기본 테이블에 대한 질의로 변환한다.
- 기본 테이블에 대한 질의를 통해 데이터를 검색한다.
- 검색된 결과를 출력한다.

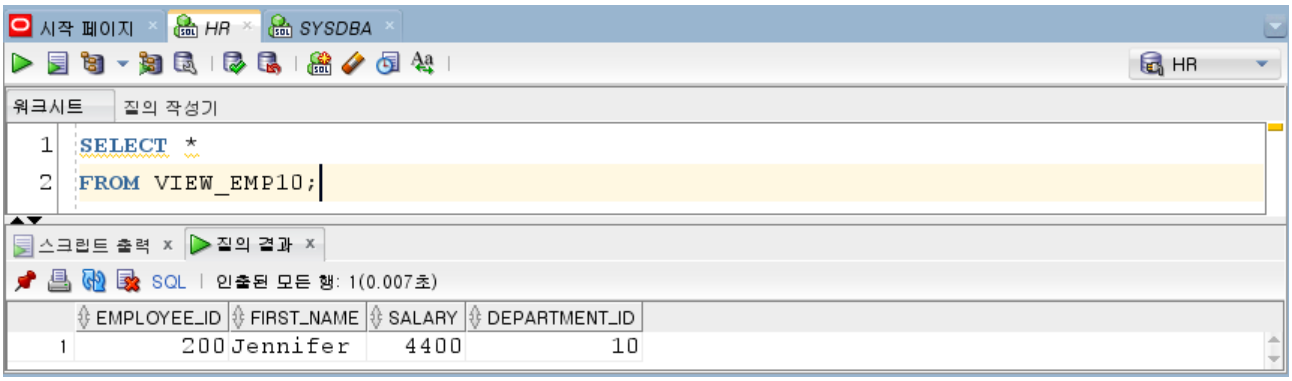
4) 뷰의 종류(뷰를 정의하기 위해서 사용되는 기본 테이블의 수에 따라 단순 뷰와 복합 뷰로 나뉜다)

단순 뷰	복합 뷰
하나의 테이블로 생성	여러 개의 테이블로 생성
그룹 함수의 사용이 불가능	그룹 함수의 사용이 가능
DISTINCT 사용이 불가능	DISTINCT 사용이 가능
DML(INSERT/UPDATE/DELETE) 사용 가능	DML(INSERT/UPDATE/DELETE) 사용 불가능

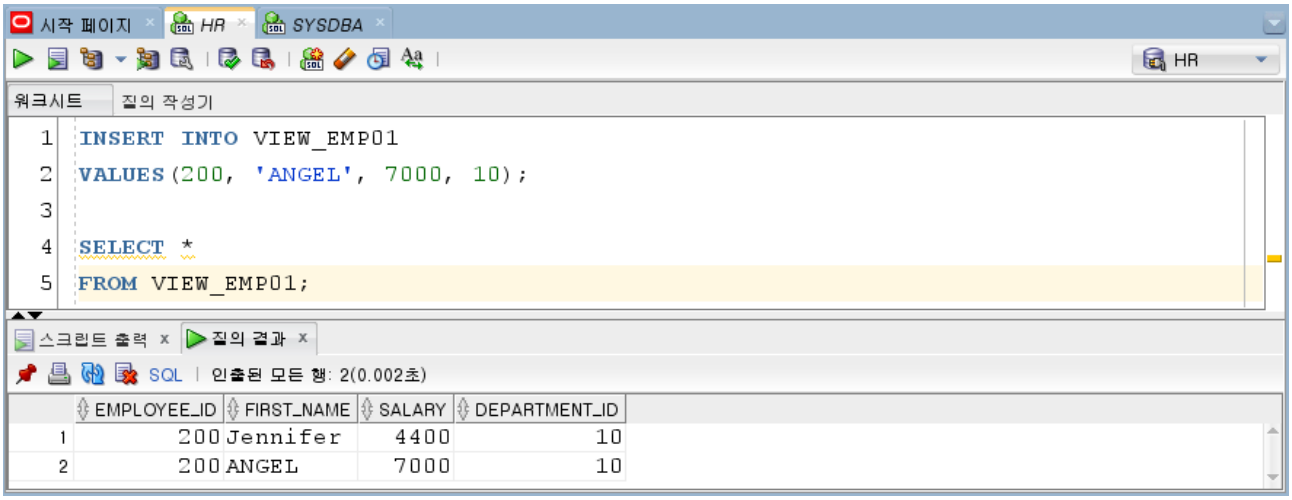
① 단순 뷰에 대한 데이터 조작

단순 뷰에서는 INSERT/UPDATE/DELETE 문을 사용할 수 있다.

```
SELECT *
FROM VIEW_EMP01;
```

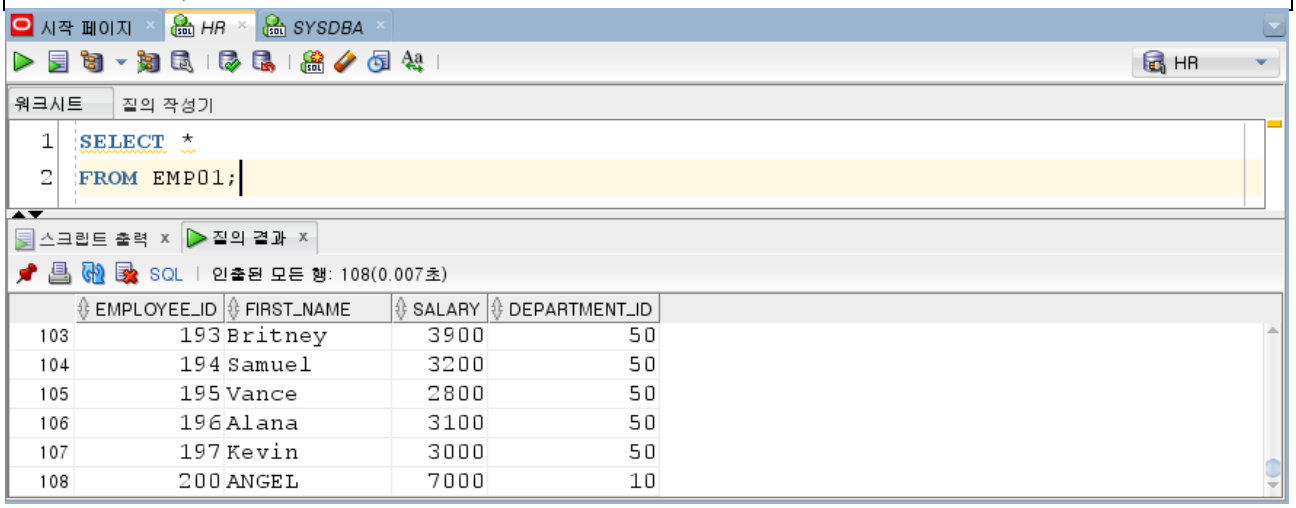



```
INSERT INTO VIEW_EMP01
VALUES(8000, 'ANGEL', 7000, 10);

SELECT *
FROM VIEW_EMP01;
```



단순 뷰를 대상으로 실행한 DML 명령문의 치리 결과는 뷰를 정의할 때 사용한 기본 테이블에 적용된다.

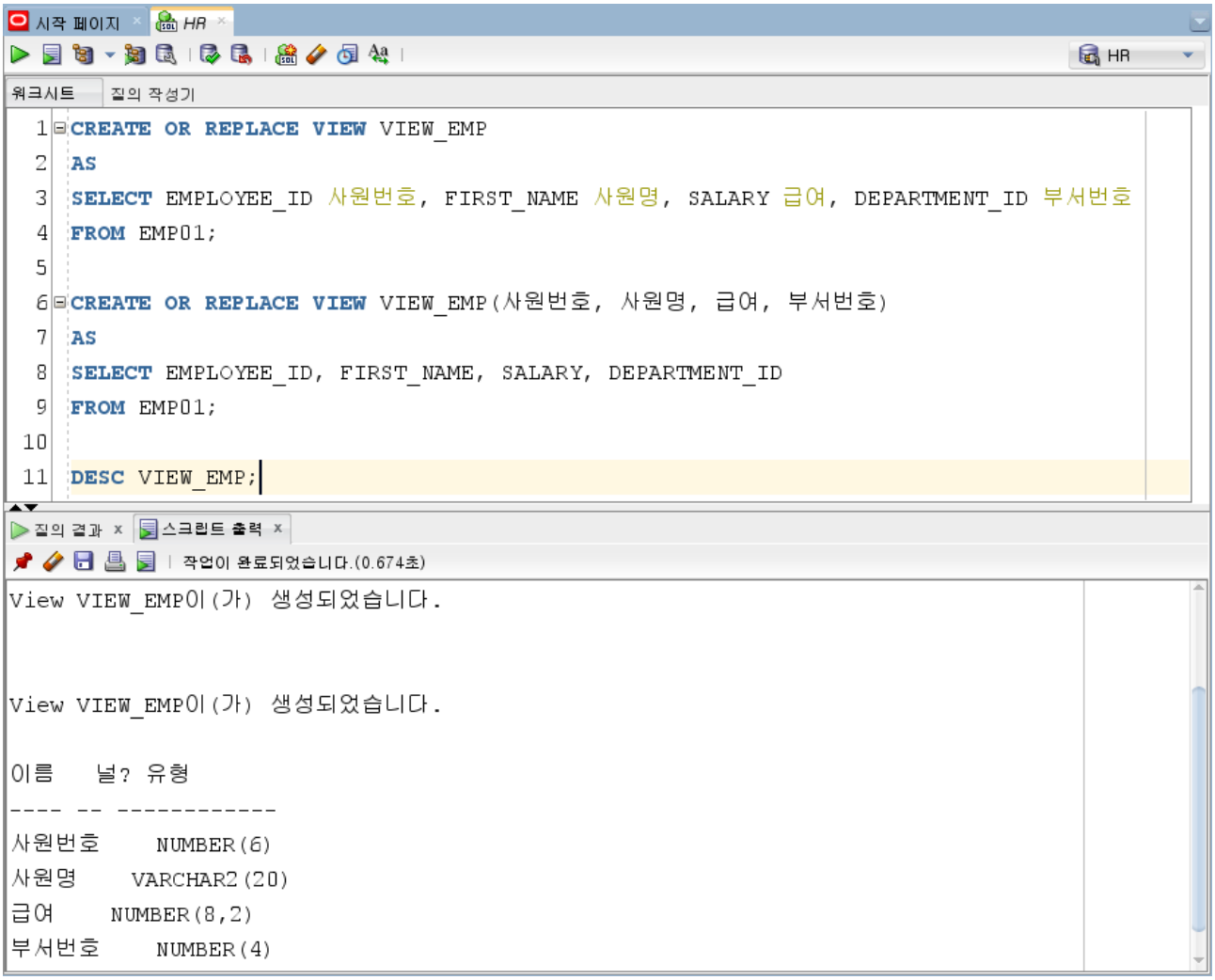
```
SELECT *
FROM EMP01;
```



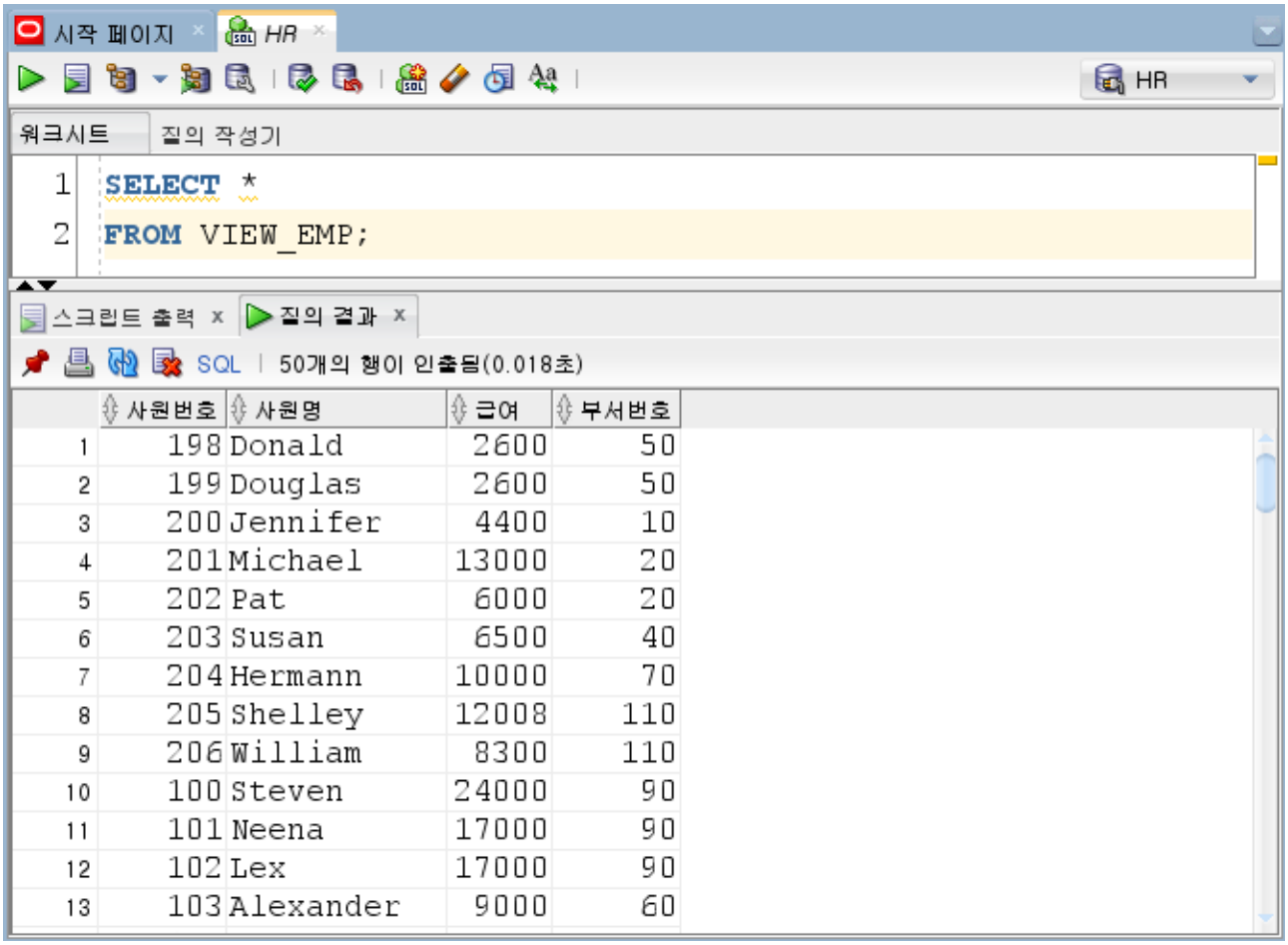
② 단순 뷰의 칼럼에 별칭 부여하기

사원번호, 사원명, 급여, 부서번호로 구성된 뷰를 작성하되 기본 테이블은 EMP01로 하고 칼럼명은 한글화 한다.

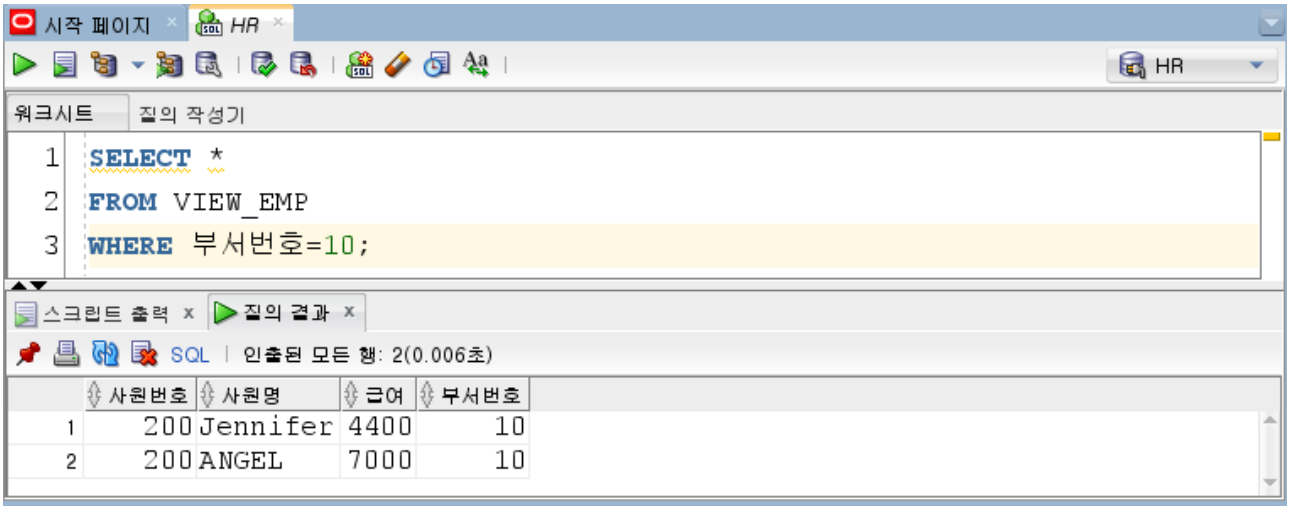
```
CREATE OR REPLACE VIEW VIEW_EMP
AS
SELECT EMPLOYEE_ID 사원번호, FIRST_NAME 사원명, SALARY 급여, DEPARTMENT_ID 부서번호
FROM EMP01;

CREATE OR REPLACE VIEW VIEW_EMP(사원번호, 사원명, 급여, 부서번호)
AS
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMP01;

DESC VIEW_EMP;
```




```
SELECT *
FROM VIEW_EMP;
```



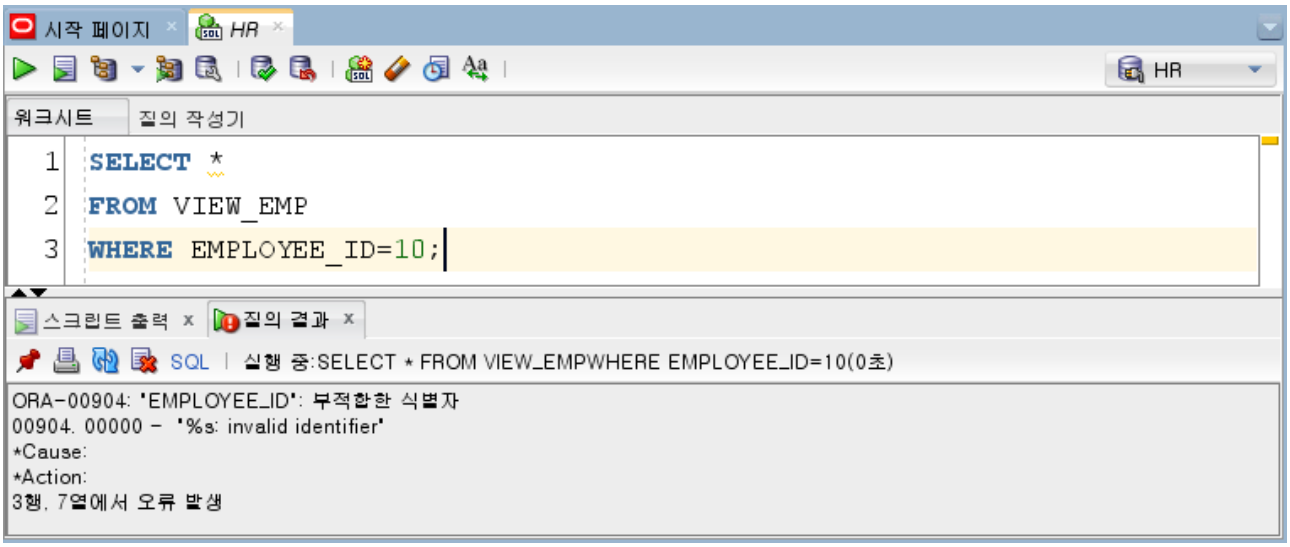

```
SELECT *
FROM VIEW_EMP
WHERE  부서번호=10;
```



칼럼의 별칭을 사용해서 뷰를 생성하면 VIEW_EMP의 칼럼 이름만 별칭으로 데이터 구조에 반영되고 EMP01 테이블의 칼럼 이름에는 전혀 영향을 주지 못한다. 오류 발생

```
SELECT *
```

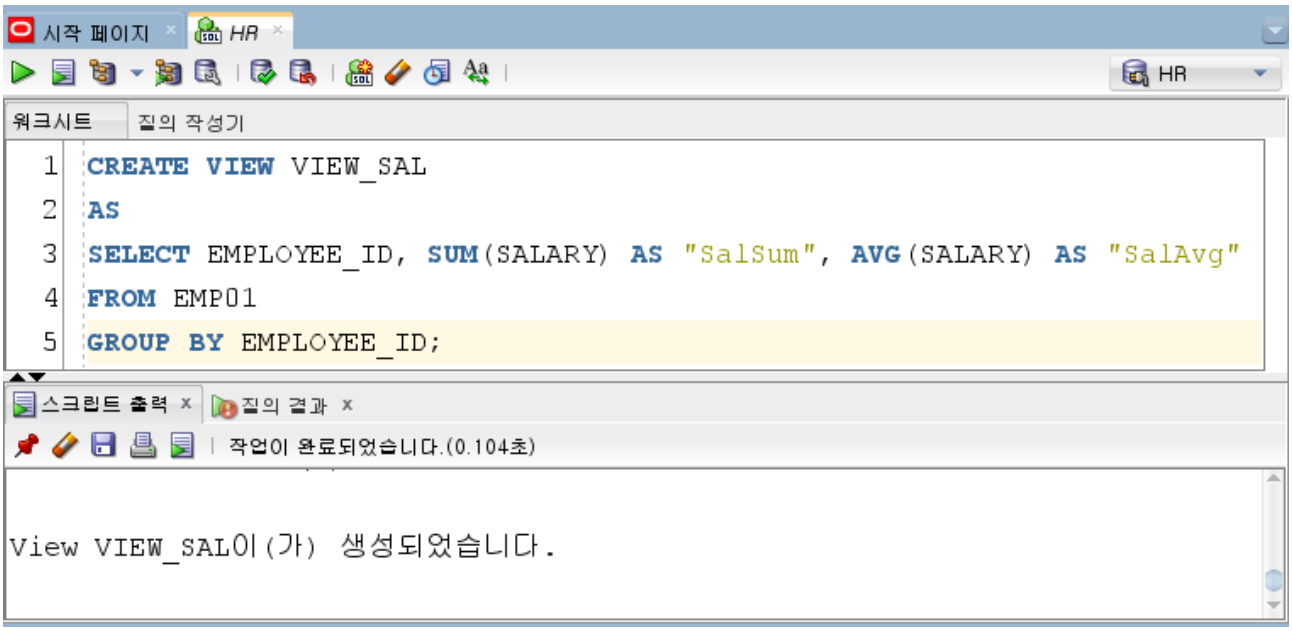
```
FROM VIEW_EMP
WHERE EMPLOYEE_ID=10;
```



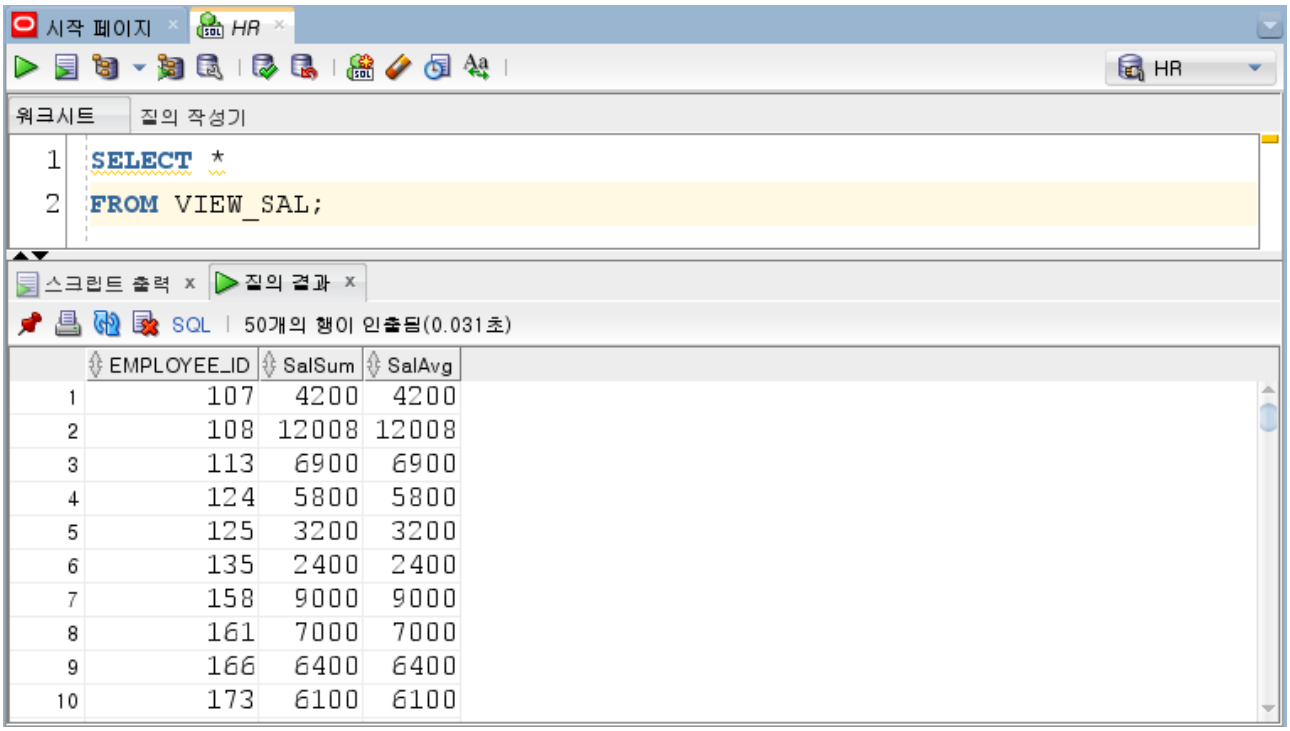
③ 그룹 함수를 사용한 단순 뷰

부서별 급여 총액과 평균을 구하는 작업을 자주 한다면 이를 뷰로 생성해 놓고 가져다 사용하면 편리하다.

```
CREATE VIEW VIEW_SAL
AS
SELECT EMPLOYEE_ID, SUM(SALARY) AS "SalSum", AVG(SALARY) AS "SalAvg"
FROM EMP01
GROUP BY EMPLOYEE_ID;
```



```
SELECT *
FROM VIEW_SAL;
```



	EMPLOYEE_ID	SalSum	SalAvg
1	107	4200	4200
2	108	12008	12008
3	113	6900	6900
4	124	5800	5800
5	125	3200	3200
6	135	2400	2400
7	158	9000	9000
8	161	7000	7000
9	166	6400	6400
10	173	6100	6100

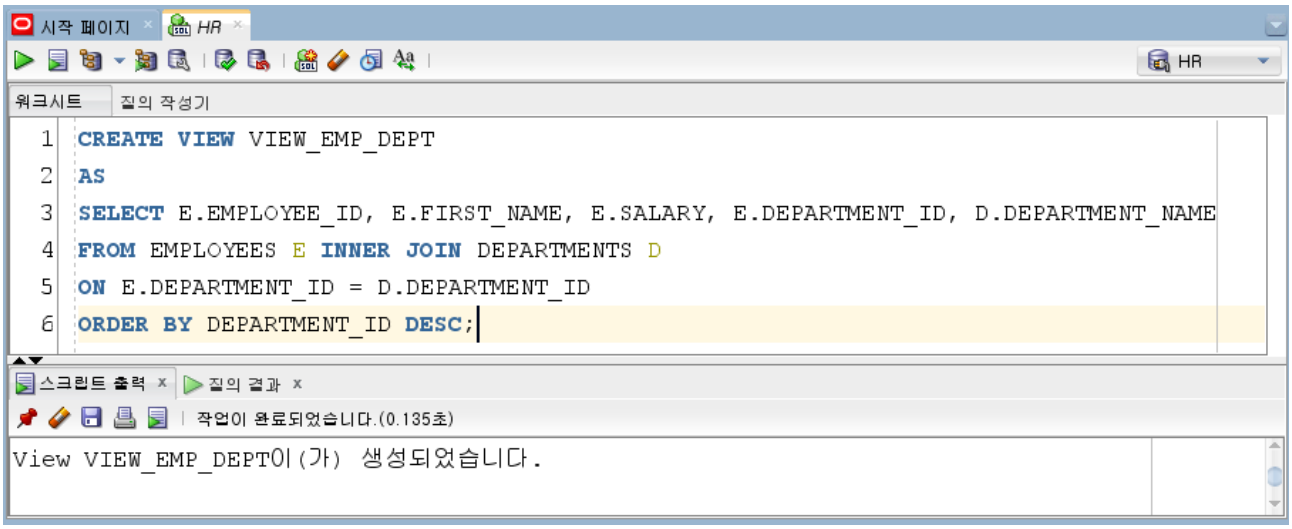
급여 총액은 SUM이란 그룹 함수의 결과는 물리적인 칼럼이 존재하지 않는 가상 칼럼이기에 칼럼 명도 상속 받을수 없다. **사용자가 반드시 이름을 따로 설정해야 한다.**

- 단순 뷰에 DML 명령어로 조작 불가능한 경우
  - 뷰 정의에 포함되지 않은 칼럼 중에 기본 테이블의 칼럼이 NOT NULL 제약 조건이 지정되어 있는 경우 INSERT 문이 사용 불가능하다.
  - SAL*12와 같이 산술 표현식으로 정의된 가상 칼럼이 뷰에 정의되면 INSERT나 UPDATE가 불가능하다.
  - DISTINCT을 포함한 경우에도 DML 명령을 사용할 수 없다.
  - 그룹 함수나 나 GROUP BY 절을 포함한 경우에도 DML 명령을 사용할 수 없다.

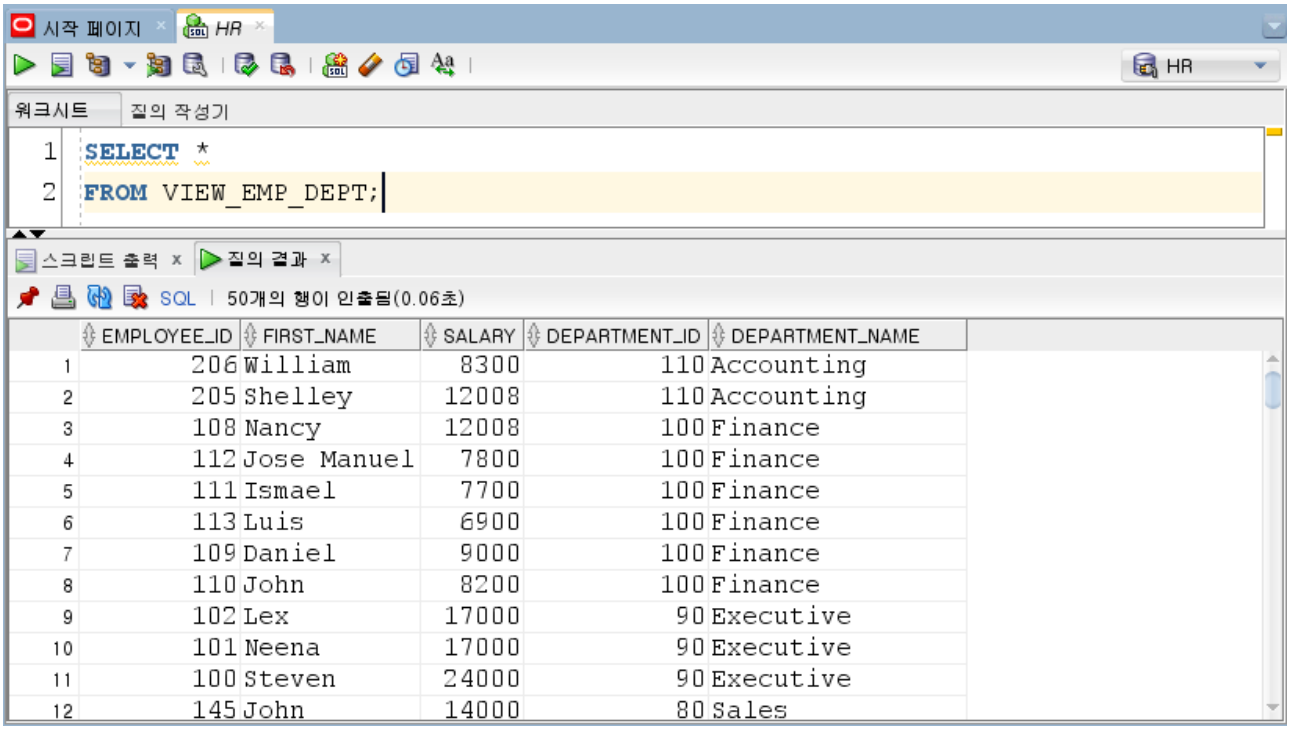
④ 복합 뷰

두 개 이상의 기본 테이블에 의해 정의한 뷰

```
CREATE VIEW VIEW_EMP_DEPT
AS
SELECT E.EMPLOYEE_ID, E.FIRST_NAME, E.SALARY, E.DEPARTMENT_ID, D.DEPARTMENT_NAME
FROM EMPLOYEES E INNER JOIN DEPARTMENTS D
ON E.DEPARTMENT_ID = D.DEPARTMENT_ID
ORDER BY DEPARTMENT_ID DESC;
```




```
SELECT *
FROM VIEW_EMP_DEPT;
```

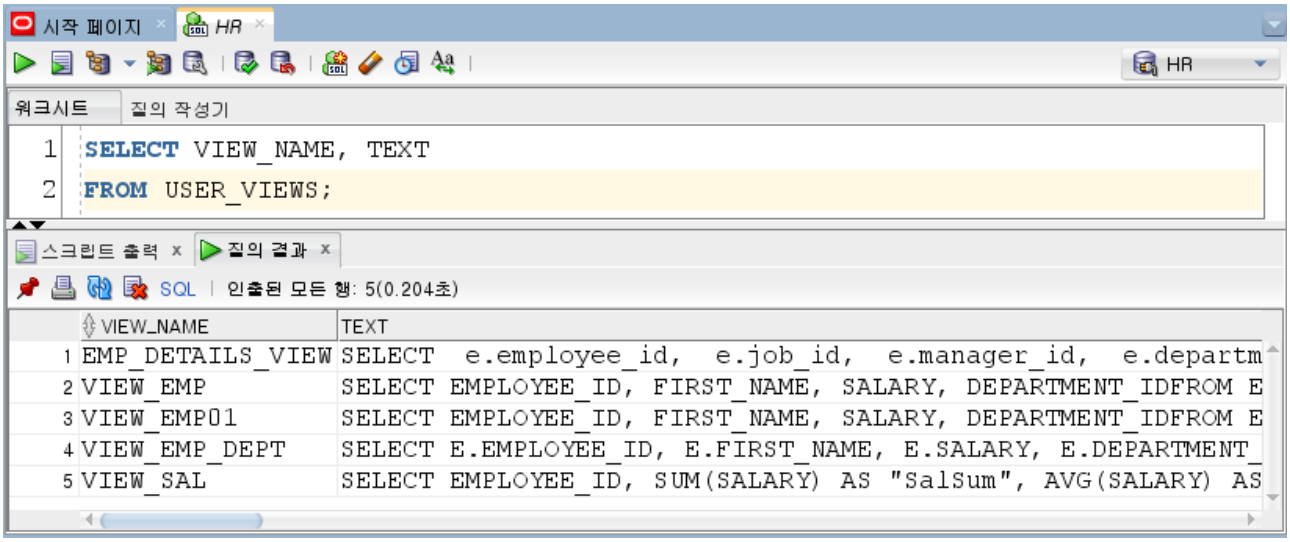


	EMPLOYEE_ID	FIRST_NAME	SALARY	DEPARTMENT_ID	DEPARTMENT_NAME
1	206	William	8300	110	Accounting
2	205	Shelley	12008	110	Accounting
3	108	Nancy	12008	100	Finance
4	112	Jose Manuel	7800	100	Finance
5	111	Ismael	7700	100	Finance
6	113	Luis	6900	100	Finance
7	109	Daniel	9000	100	Finance
8	110	John	8200	100	Finance
9	102	Lex	17000	90	Executive
10	101	Neena	17000	90	Executive
11	100	Steven	24000	90	Executive
12	145	John	14000	80	Sales

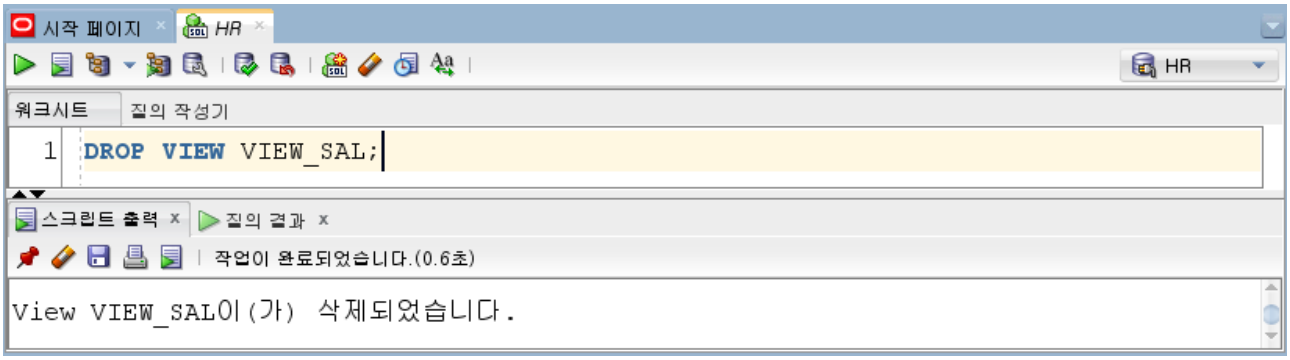
5) 뷰 삭제

뷰는 실체가 없는 가상 테이블이기 때문에 뷰를 삭제한다는 것은
USER_VIEWS 데이터 딕셔너리에 저장되어 있는 뷰의 정의를 삭제하는 것을 의미한다.

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




```
DROP VIEW VIEW_SAL;
```



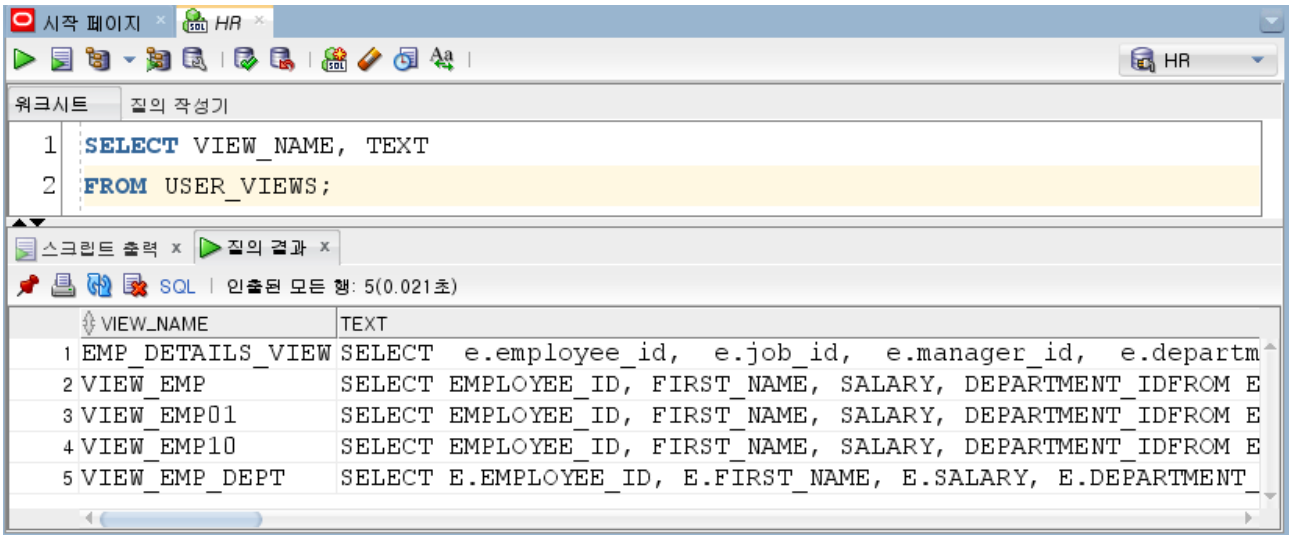

```
SELECT VIEW_NAME, TEXT
FROM USER_VIEWS;
```

6) 뷰 수정을 위한 OR REPLACE 옵션

CREATE VIEW 대신 CREATE OR REPLACE VIEW를 사용하면 존재하지 않은 뷰이면 새로운 뷰를 생성하고 기존에 존재하는 뷰이면 그 내용을 변경한다.

```
CREATE OR REPLACE VIEW VIEW_EMP10
AS
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMP01
WHERE EMPLOYEE_ID=10;
```

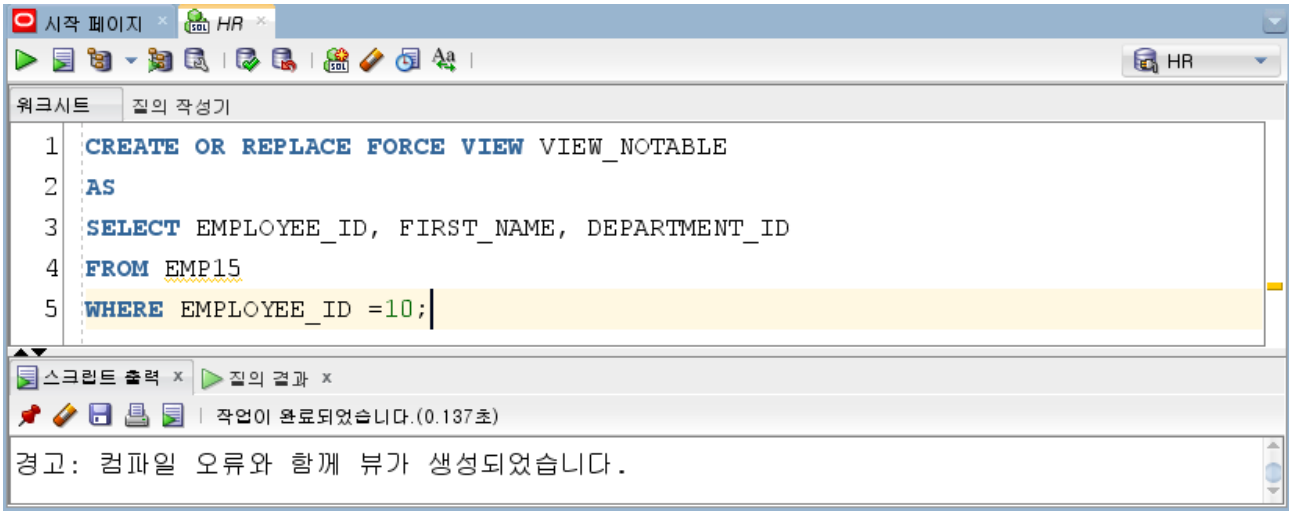
```
SELECT VIEW_NAME, TEXT
FROM USER_VIEWS;
```



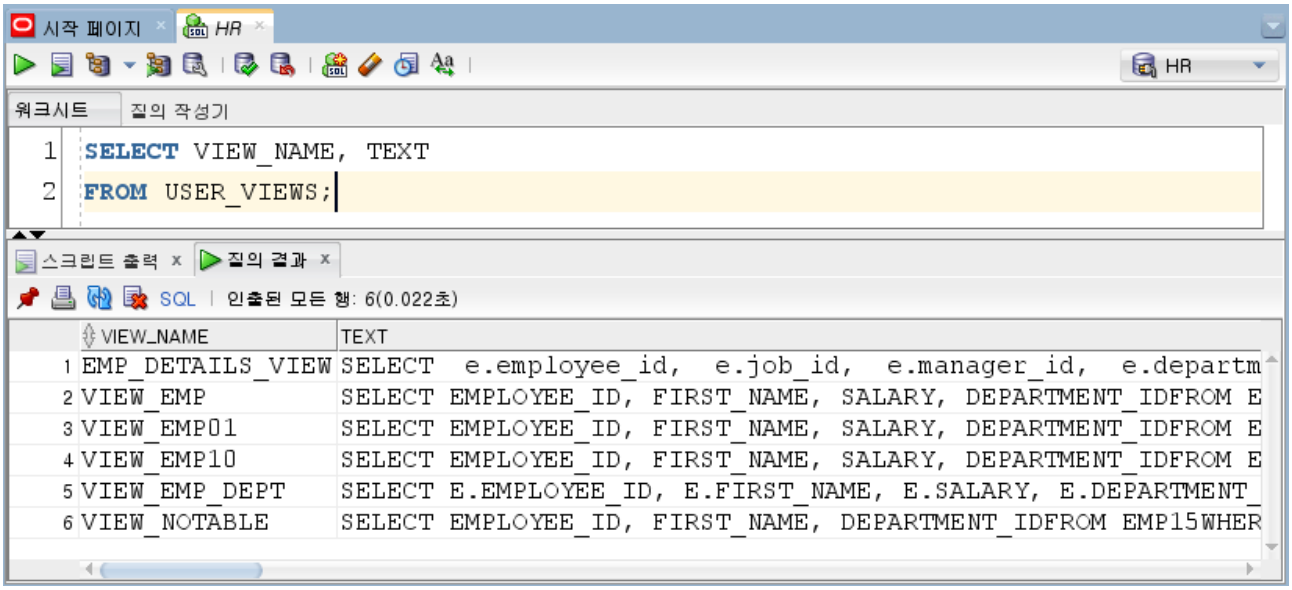
7) 기본 테이블 없이 뷰를 생성하기 위한 FORCE 옵션

기본 테이블이 존재하지 않더라도 뷰를 생성하려면 FORCE 옵션을 추가해야 한다.

```
CREATE OR REPLACE FORCE VIEW VIEW_NOTABLE
AS
SELECT EMPLOYEE_ID, FIRST_NAME, DEPARTMENT_ID
FROM EMP15
WHERE EMPLOYEE_ID =10;
```




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

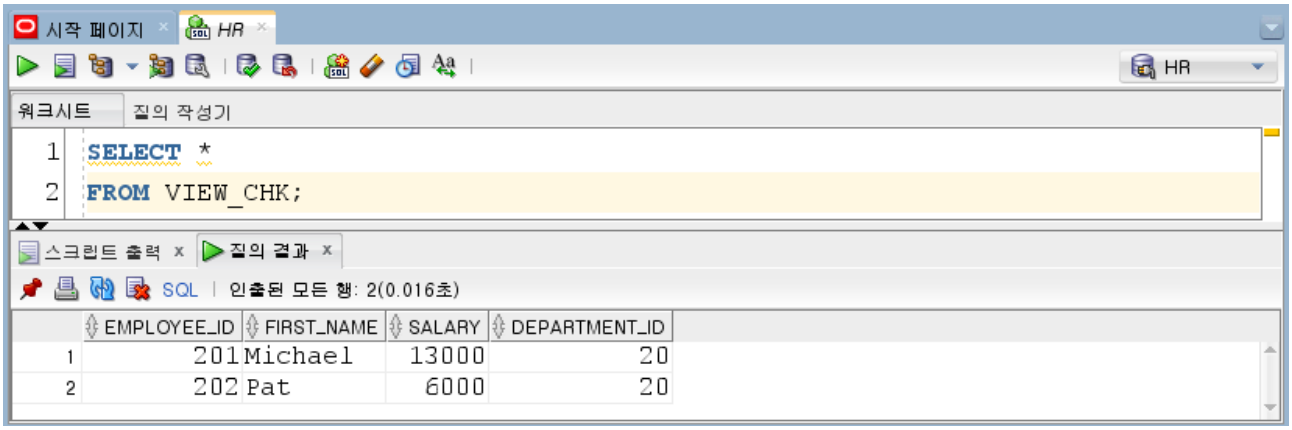


8) WITH CHECK OPTION

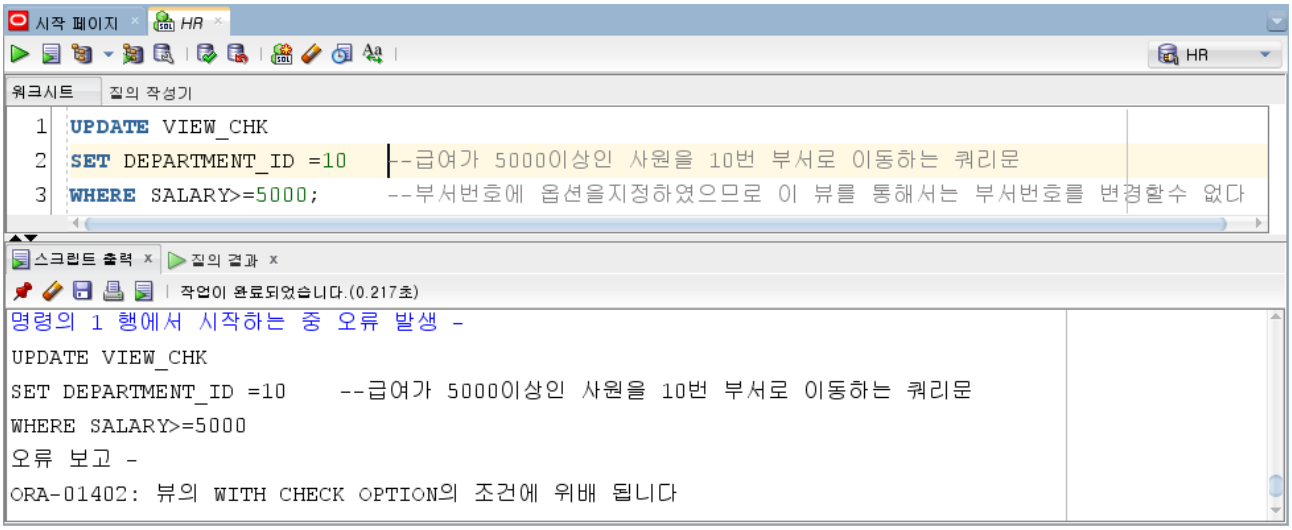
WITH CHECK OPTION 옵션은 뷰 생성시 조건으로 지정한 칼럼 값을 변경하지 못하도록 하는 것이다.

```
CREATE OR REPLACE VIEW VIEW_CHK
AS
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMP01
WHERE DEPARTMENT_ID=20 WITH CHECK OPTION;
```

```
SELECT *
FROM VIEW_CHK;
```




```
UPDATE VIEW_CHK
SET DEPARTMENT_ID =10  --급여가 5000이상인 사원을 10번 부서로 이동하는 쿼리문
WHERE SALARY>=5000;  --부서번호에 옵션을지정하였으므로 이 뷰를 통해서는 부서번호를 변경할수 없다
```

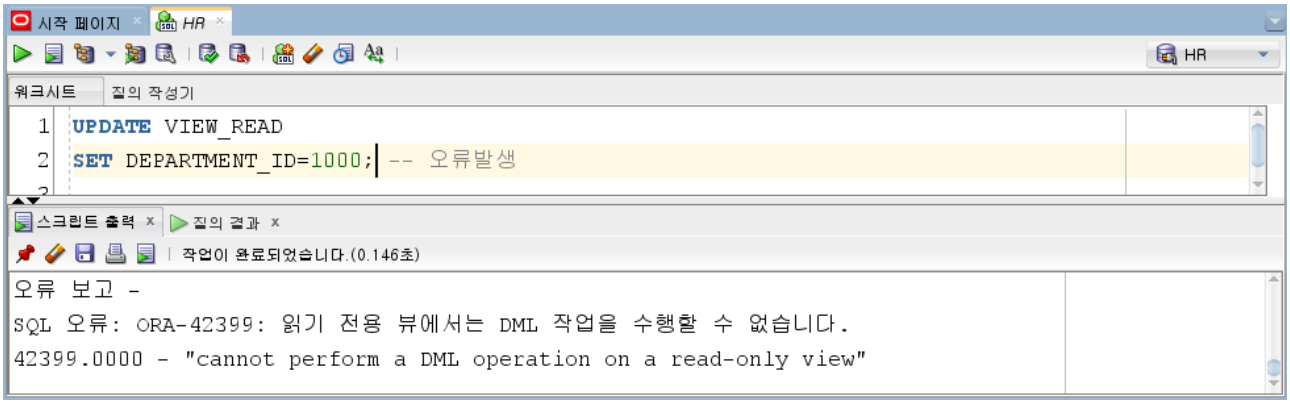


9) WITH READ ONLY

뷰를 통해서는 기본 테이블의 어떤 컬럼에 대해서도 내용을 절대 변경할 수 없도록 하는 것이다.

```
CREATE OR REPLACE VIEW VIEW_READ
AS
SELECT EMPLOYEE_ID, FIRST_NAME, SALARY, DEPARTMENT_ID
FROM EMP01
WHERE DEPARTMENT_ID=30 WITH READ ONLY;
```

```
UPDATE VIEW_READ
SET DEPARTMENT_ID=1000; -- 오류발생
```



- **WITH CHECK OPTION**은 조건에 사용한 컬럼의 값을 수정하지 못하게 하는 것이고,
- **WITH READ ONLY**는 기본 테이블의 모두를 수정하지 못하게 하는 것이다.

10) 뷰 활용하기

사원 중에서 입사일이 빠른 사람 5명(TOP-5)만을 얻어 오는 질의문을 작성. **ROWNUM** 칼럼을 이용한다. ROWNUM칼럼은 오라클에서 내부적으로 부여되는데 INSERT문에 의해 입력한 순서에 따라 1씩 증가되면서 값이 지정된다.

```
SELECT ROWNUM, EMPLOYEE_ID, FIRST_NAME, HIRE_DATE
FROM EMPLOYEES;
```

	ROWNUM	EMPLOYEE_ID	FIRST_NAME	HIRE_DATE
1	1	198	Donald	07/06/21
2	2	199	Douglas	08/01/13
3	3	200	Jennifer	03/09/17
4	4	201	Michael	04/02/17
5	5	202	Pat	05/08/17
6	6	203	Susan	02/06/07
7	7	204	Hermann	02/06/07
8	8	205	Shelley	02/06/07
9	9	206	William	02/06/07
10	10	100	Steven	03/06/17
11	11	101	Neena	05/09/21
12	12	102	Lex	01/01/13
13	13	103	Alexander	06/01/03
14	14	104	Bruce	07/05/21
15	15	105	David	05/06/25
16	16	106	Valli	06/02/05

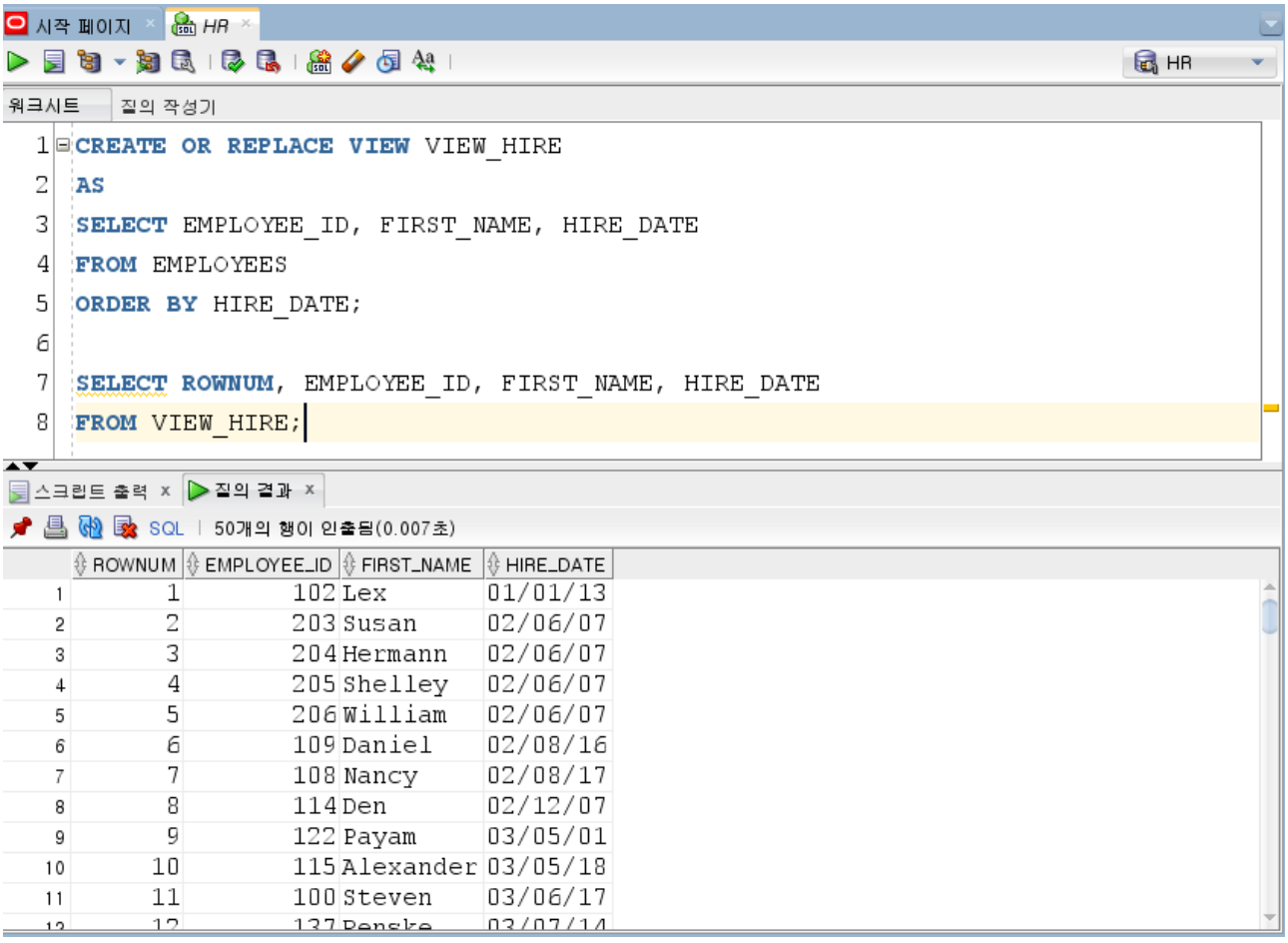
```
SELECT ROWNUM, EMPLOYEE_ID, FIRST_NAME, HIRE_DATE
FROM EMPLOYEES
ORDER BY HIRE_DATE;
```

	ROWNUM	EMPLOYEE_ID	FIRST_NAME	HIRE_DATE
1	12	102	Lex	01/01/13
2	6	203	Susan	02/06/07
3	7	204	Hermann	02/06/07
4	8	205	Shelley	02/06/07
5	9	206	William	02/06/07
6	19	109	Daniel	02/08/16
7	18	108	Nancy	02/08/17
8	24	114	Den	02/12/07
9	32	122	Payam	03/05/01
10	25	115	Alexander	03/05/18
11	10	100	Steven	03/06/17
12	47	137	Renske	03/07/14

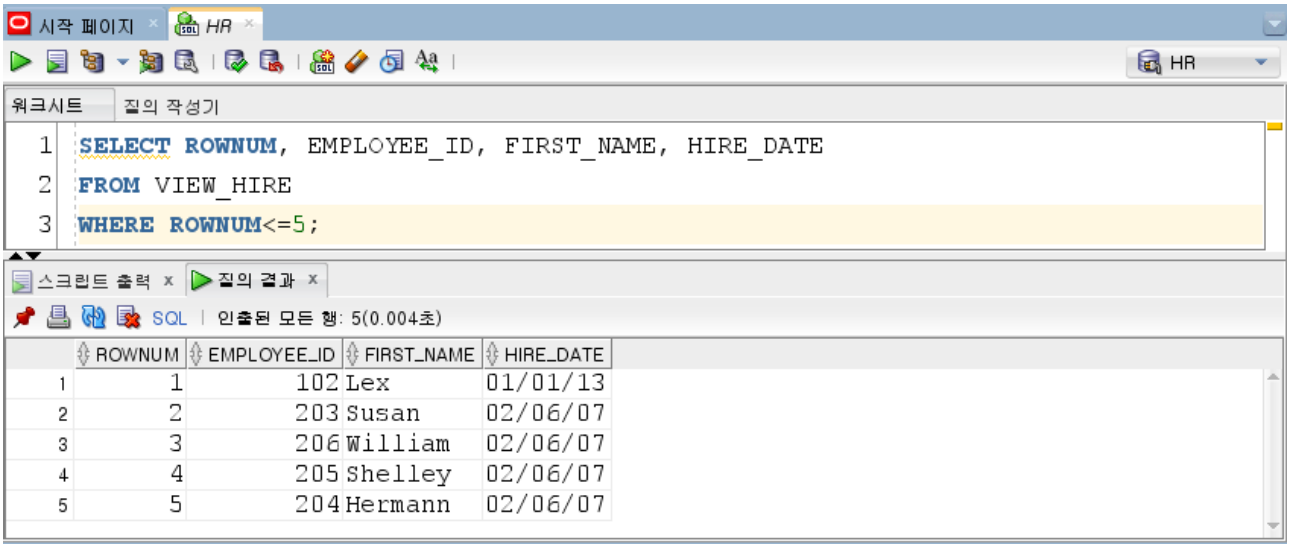
입사일을 기준으로 오름차순으로 정렬을 하였는데도 해당 행의 ROWNUM은 바뀌지 않는다.
데이터가 입력된 시점에서 결정되면 다시는 값이 바뀌지 않기 때문이다.
새로운 테이블에 입사일을 기준으로 오름차순 정렬한 쿼리문의 결과를 저장하면 입사일이 가장 빠른 사원이 제일 처음으로 입력되므로 ROWNUM 컬럼 값이 1부터 부여된다.

```
CREATE OR REPLACE VIEW VIEW_HIRE
AS
SELECT EMPLOYEE_ID, FIRST_NAME, HIRE_DATE
FROM EMPLOYEES
ORDER BY HIRE_DATE;
```

```
SELECT ROWNUM, EMPLOYEE_ID, FIRST_NAME, HIRE_DATE
FROM VIEW_HIRE;
```



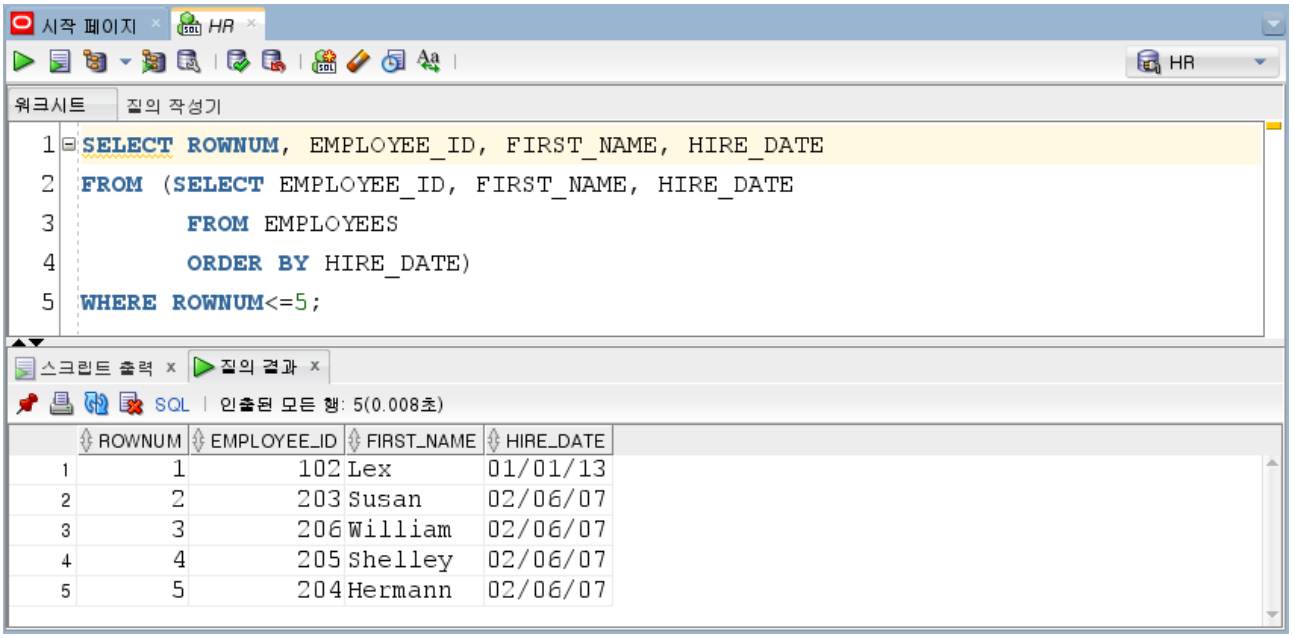

```
SELECT ROWNUM, EMPLOYEE_ID, FIRST_NAME, HIRE_DATE
FROM VIEW_HIRE
WHERE ROWNUM<=5;
```



① 인라인 뷰로 TOP-N 구하기

인라인 뷰는 SQL 문장에서 사용하는 서브 쿼리의 일종으로 보통 FROM 절에 위치해서 테이블처럼 사용하는 것이다. 인라인 뷰란 메인 쿼리의 SELECT 문의 FROM 절 내부에 사용된 서브 쿼리문을 말한다. CREATE VIEW로 생성하는 것이 아니라 인라인 뷰는 SQL문 내부에 뷰를 정의하고 이를 테이블처럼 사용한다.

```
SELECT ROWNUM, EMPLOYEE_ID, FIRST_NAME, HIRE_DATE
FROM (SELECT EMPLOYEE_ID, FIRST_NAME, HIRE_DATE
      FROM EMPLOYEES
      ORDER BY HIRE_DATE)
WHERE ROWNUM<=5;
```



[문제]

1. 사원 번호와 사원명과 부서명과 부서의 위치를 출력하는 뷰(VIEW_LOC)를 작성하라.

2. 30번 부서 소속 사원의 이름과 입사일과 부서명을 출력하는 뷰(VIEW_DEPT30)를 작성하라.

3. 부서별 최대 급여 정보를 가지는 뷰(VIEW_DEPT_MAXSAL)를 생성하라.

4. 급여를 많이 받는 순서대로 3명만 출력하는 뷰(VIEW_SAL_TOP3)와 인라인 뷰로 작성하라.