

8. 무결성 제약 조건

잘못된 데이터가 입력되지 않도록 무결성 제약 조건을 지정한다.

NULL을 허용하지 않도록 하려면 NOT NULL 제약 조건을 지정한다.

항상 유일해야하는 고유 키를 지정하려면 UNIQUE 제약 조건을 설정한다.

칼럼값은 반드시 존재해야 하고 유일하도록 하려면 PRIMARY KEY 제약 조건을 설정한다.

해당 칼럼값은 참조되는 테이블의 칼럼에 하나 이상과 일치하도록 하려면 FOREIGN KEY 제약 조건을 설정한다.

1) 무결성 제약 조건의 개념과 종류

무결성 제약 조건은 데이터를 추가, 수정, 삭제하는 과정에서 무결성을 유지할 수 있도록 제약을 주는 것을 말한다. 데이터 무결성이란 데이터베이스 내의 데이터에 대한 정확성, 일관성, 유효성, 신뢰성을 보장하기 위해 데이터 변경 혹은 수정 시 여러가지 제한을 두어 데이터의 정확성을 보증하는 것을 말한다. 관계형 데이터베이스 관리시스템(RDBMS)에서는 데이터베이스 설계 시 무결성을 고려한 설계로 데이터를 보호할 필요가 있다. 제약조건이란 바람직하지 않은 데이터가 저장되는 것을 방지하는 위해 테이블을 생성할 때 각 컬럼에 대해서 정의하는 여러 가지 규칙을 말한다.

• 무결성 제약조건

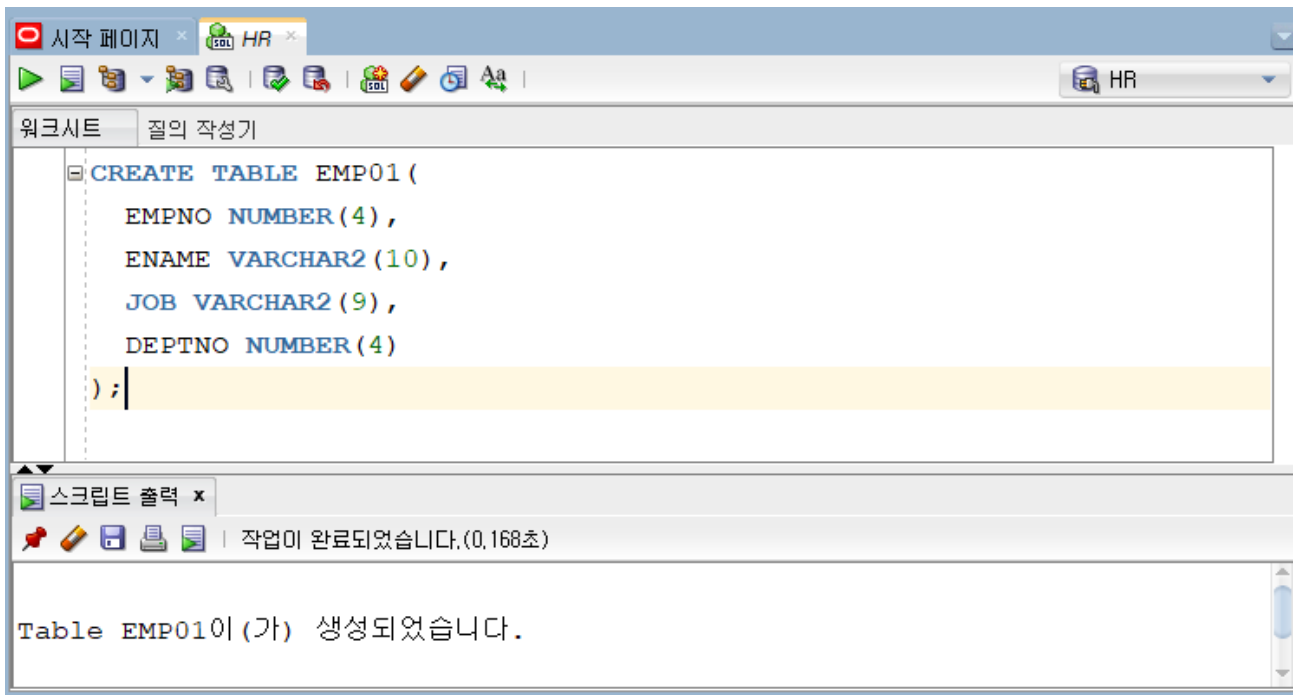
무결성 제약 조건	역	할
NOT NULL		NULL을 허용하지 않는다.
UNIQUE		중복된 값을 허용하지 않는다. 항상 유일한 값을 갖도록 한다.
PRIMARY KEY		NULL을 허용하지 않고 중복된 값을 허용하지 않는다. NOT NULL 조건과 UNIQUE 조건을 결합한 형태이다.
FOREIGN KEY		참조되는 테이블의 칼럼의 값이 존재하면 허용한다.
CHECK		저장 가능한 데이터 값의 범위나 조건을 지정하여 설정한 값만을 허용한다.

① NOT NULL 제약 조건

사원번호, 사원명, 직급, 부서번호 4개의 칼럼으로 구성된 아무런 제약조건 없이 EMP01 테이블 생성

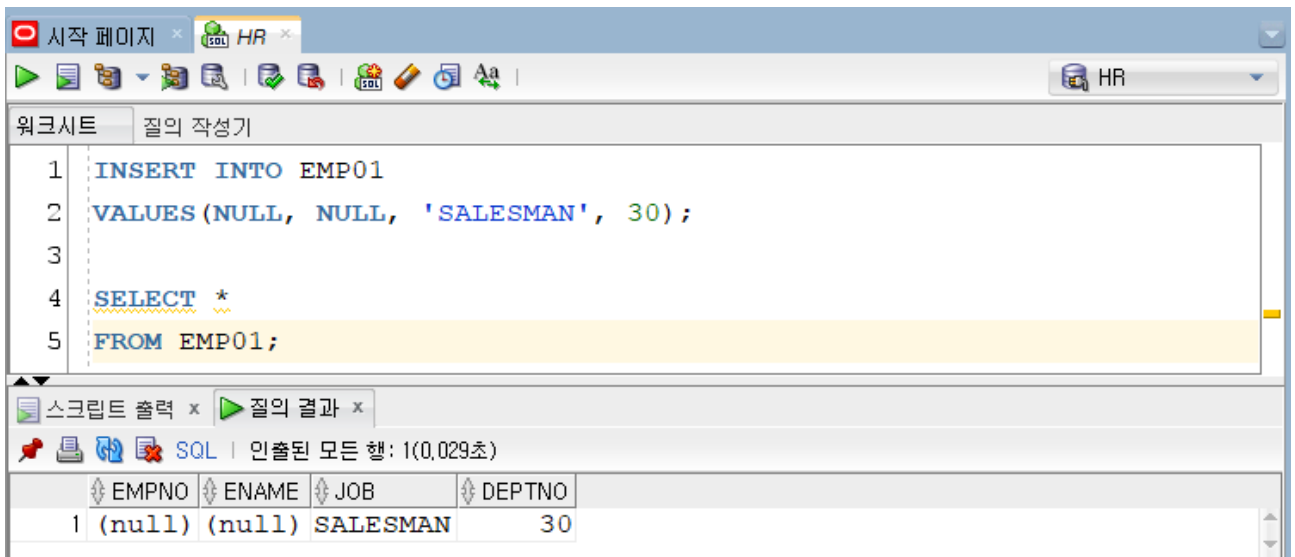
```
DROP TABLE EMP01;
```

```
CREATE TABLE EMP01(  
  EMPNO NUMBER(4),  
  ENAME VARCHAR2(10),  
  JOB VARCHAR2(9),  
  DEPTNO NUMBER(4)  
);
```



```
INSERT INTO EMP01
VALUES(NULL, NULL, 'SALESMAN', 30);

SELECT *
FROM EMP01;
```



NOT NULL 제약조건을 지정하지 않았기 때문에 NULL 값이 저장된다.

사원 테이블에 사원의 정보를 저장할 때 사원번호와 사원이름이 반드시 저장되도록 하기 위해서 사원 테이블을 생성할 때 사원번호와 사원이름을 NOT NULL 조건으로 지정해야 한다.

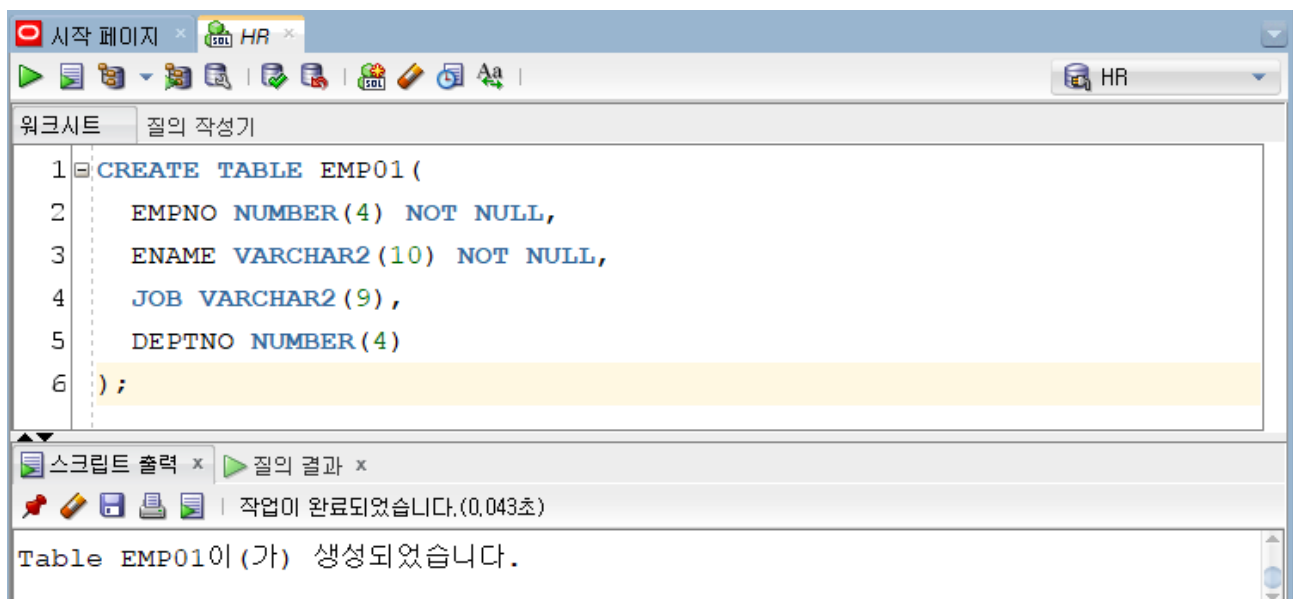
NOT NULL 제한 조건은 해당 칼럼에 NULL 값을 추가하거나 NULL 값으로 변경하는 것을 막는다.

제약 조건은 칼럼명과 자료형을 기술한 후에 연이어서 NOT NULL을 기술하면 된다.

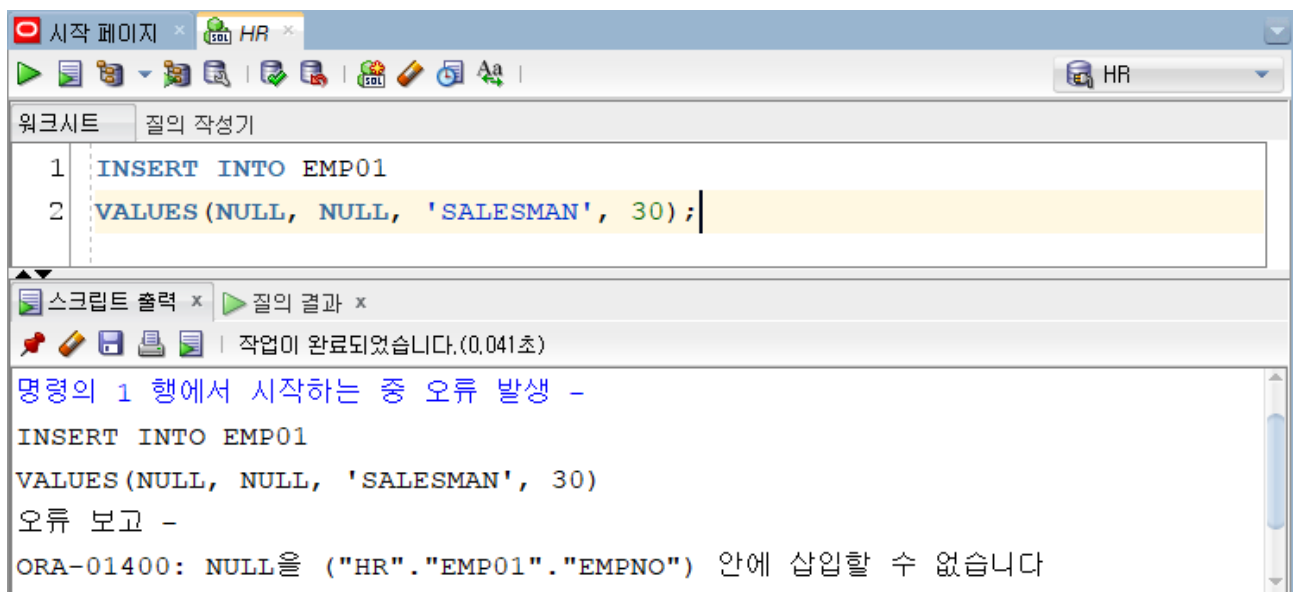
사원번호, 사원명, 직급, 부서번호 4개의 칼럼으로 구성하되 사원번호와 사원명에 NOT NULL 조건을 지정하여 EMP01 테이블 생성

```
DROP TABLE EMP01;
```

```
CREATE TABLE EMP01(  
  EMPNO NUMBER(4) NOT NULL,  
  ENAME VARCHAR2(10) NOT NULL,  
  JOB VARCHAR2(9),  
  DEPTNO NUMBER(4)  
);
```



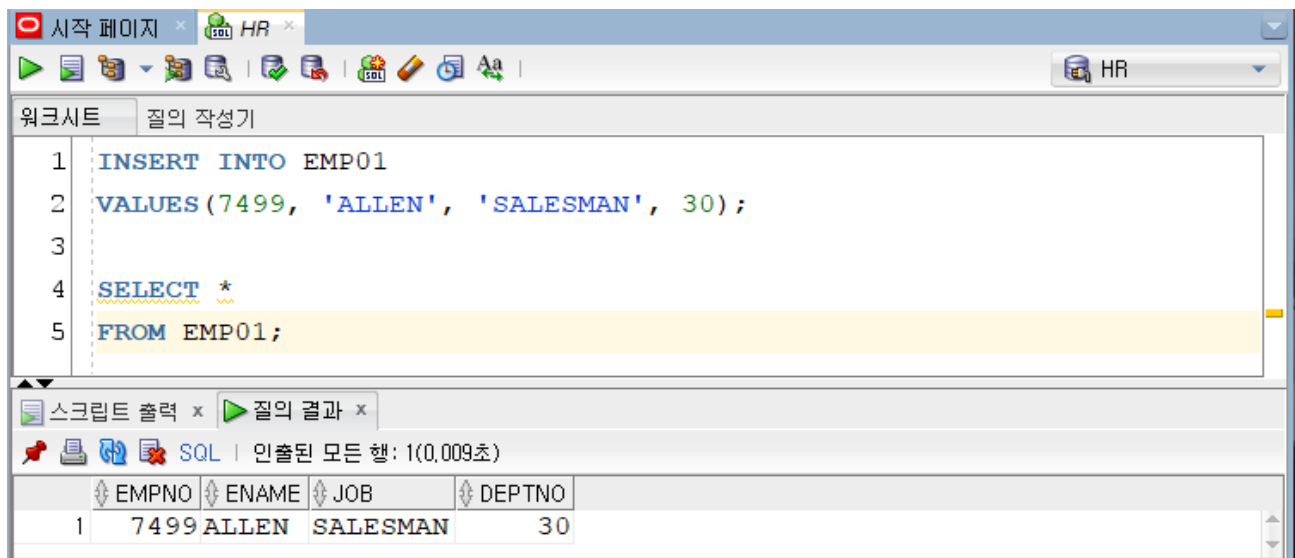
```
INSERT INTO EMP01  
VALUES(NULL, NULL, 'SALESMAN', 30);
```



NOT NULL 제약조건을 지정하였기에 직원번호와 직원명은 필수 입력란으로 NULL 값은 저장되지 못하고 오류가 발생하므로 반드시 입력해야 한다.

```
INSERT INTO EMP01
VALUES(7499, 'ALLEN', 'SALESMAN', 30);

SELECT *
FROM EMP01;
```



② UNIQUE 제약 조건

UNIQUE 제약 조건이란 특정 칼럼에 대해 자료가 중복되지 않게 하는 것이다.
즉, 지정된 칼럼에는 유일한 값이 수록되게 하는 것이다.

직원번호가 직원들을 구분하기 위한 칼럼인데도 불구하고 동일한 사번을 갖게 되면 문제가 생긴다

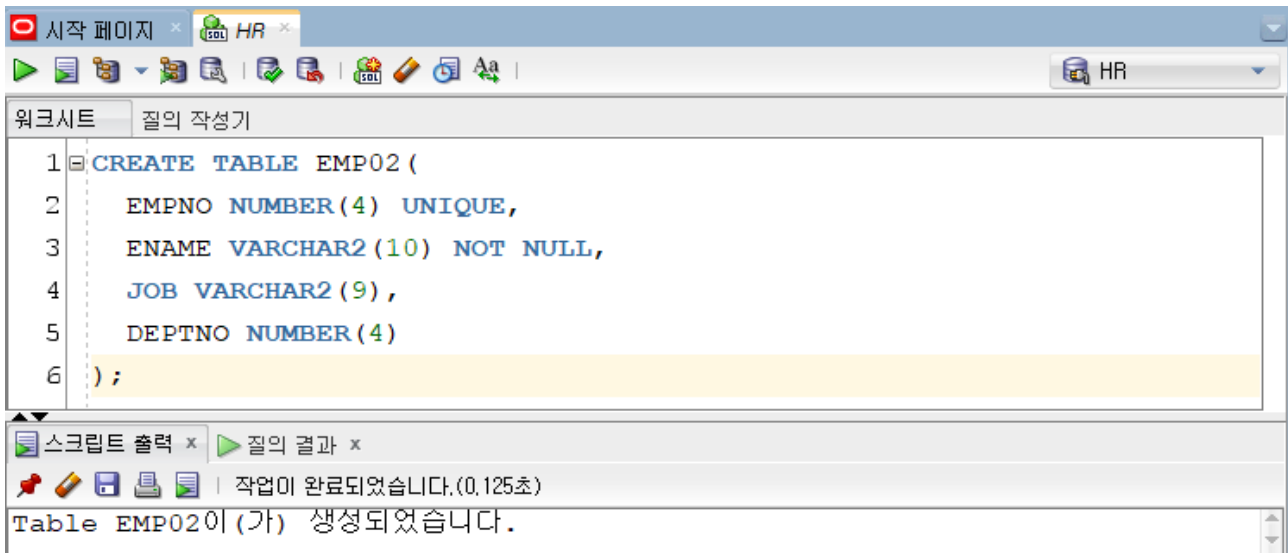
EMPNO	ENAME	JOB	DEPTNO
7499	ALLEN	SALESMAN	30
7499	JONES	SALESMAN	20

위와 같은 결과를 막고자 할 때는 UNIQUE KEY 제한조건을 사용한다.

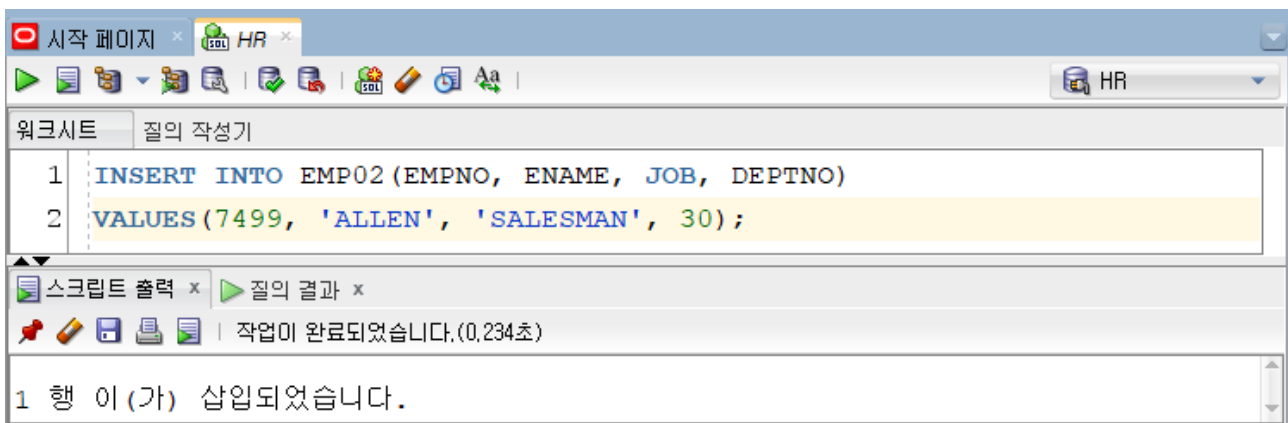
직원 테이블의 직원번호를 유일키로 지정하기 위해 제약조건은 칼럼명과 자료형을 기술한 후에 연이어서 UNIQUE를 기술하면 된다.

```
DROP TABLE EMP02;

CREATE TABLE EMP02(
  EMPNO NUMBER(4) UNIQUE,
  ENAME VARCHAR2(10) NOT NULL,
  JOB VARCHAR2(9),
  DEPTNO NUMBER(4)
);
```



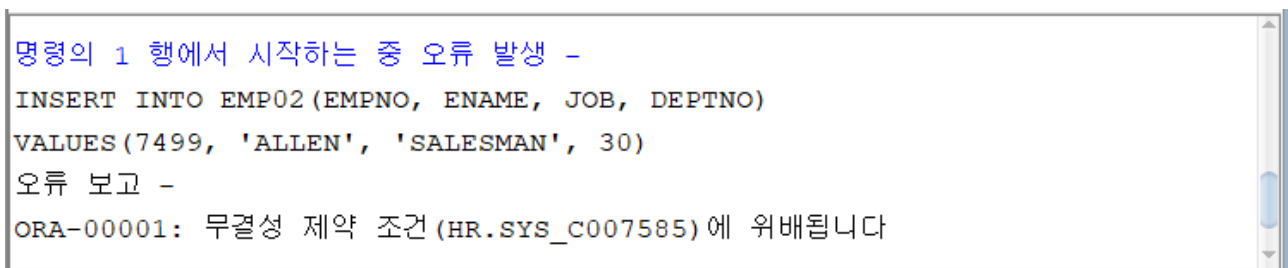
```
INSERT INTO EMP02(EMPNO, ENAME, JOB, DEPTNO)
VALUES(7499, 'ALLEN', 'SALESMAN', 30);
```



동일한 사원번호 7499번으로 입력해본다.

```
INSERT INTO EMP02(EMPNO, ENAME, JOB, DEPTNO)
VALUES(7499, 'ALLEN', 'SALESMAN', 30);
```

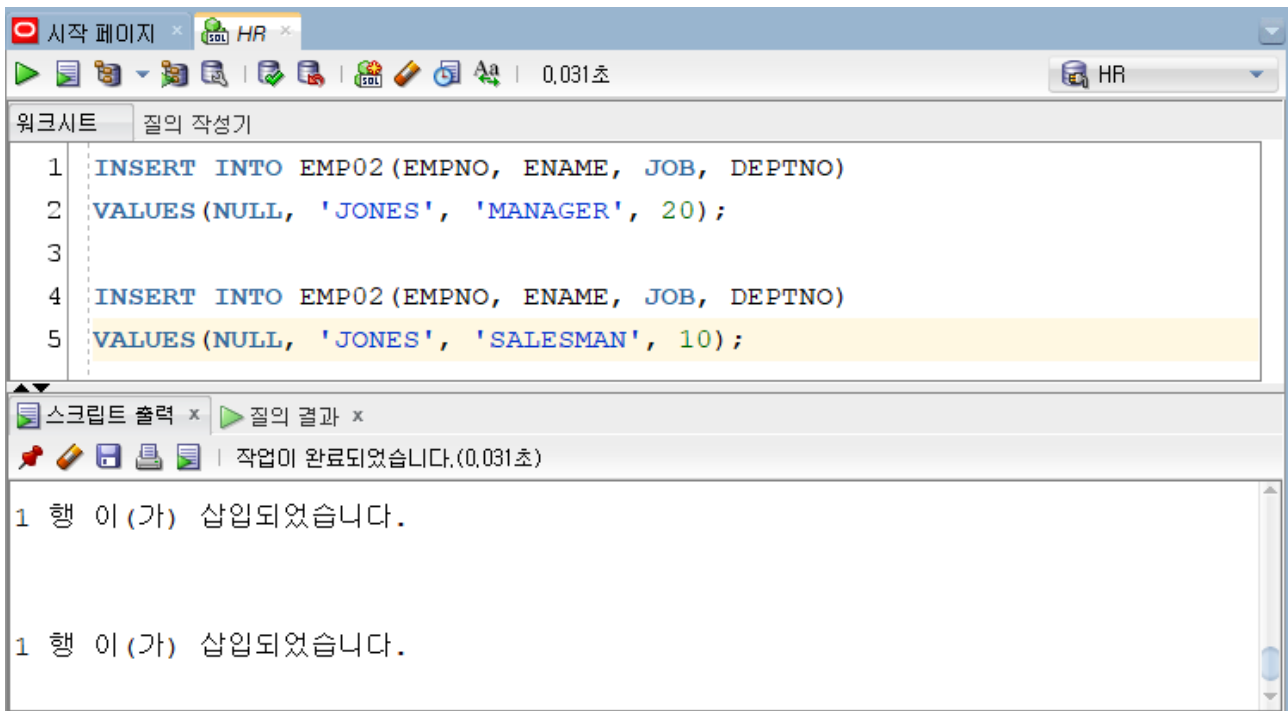
→ 오류 발생



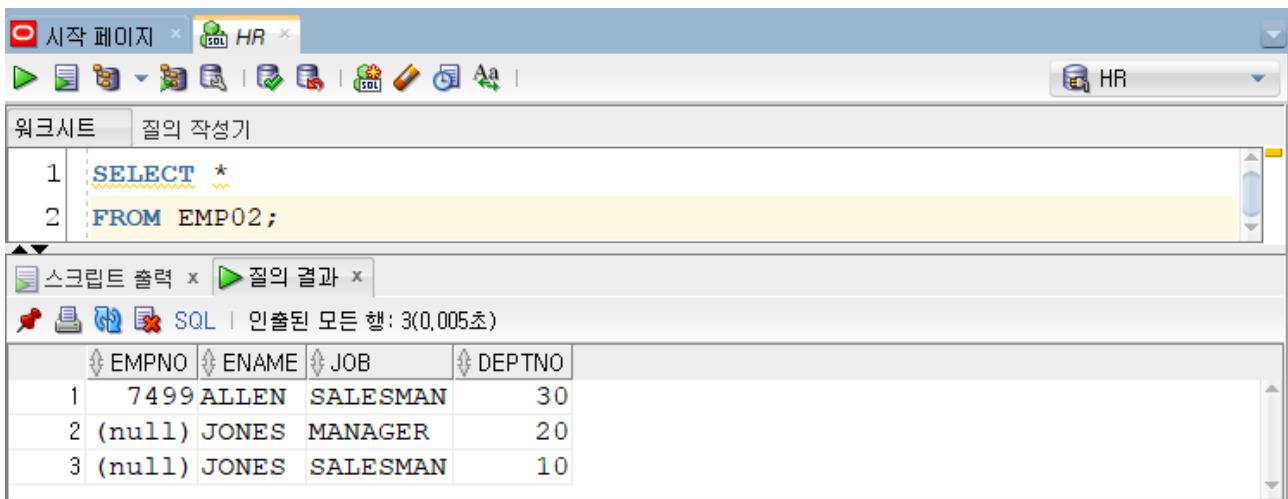
NULL은 값(VALUE)에서 제외되므로 유일한 조건인지를 체크하는 값에서 제외된다.

```
INSERT INTO EMP02(EMPNO, ENAME, JOB, DEPTNO)
VALUES(NULL, 'JONES', 'MANAGER', 20);
```

```
INSERT INTO EMP02(EMPNO, ENAME, JOB, DEPTNO)
VALUES(NULL, 'JONES', 'SALESMAN', 10);
```



```
SELECT *
FROM EMP02;
```



```
DROP TABLE DEPT01;
```

```
CREATE TABLE DEPT01 (
  DEPTNO NUMBER NOT NULL,
  DNAME VARCHAR2(14) DEFAULT NULL,
```

```


LOC VARCHAR2(13) DEFAULT NULL,
PRIMARY KEY (DEPTNO)
);

INSERT INTO DEPT01 (DEPTNO, DNAME, LOC) VALUES (10, 'ACCOUNTING', 'NEW YORK');
INSERT INTO DEPT01 (DEPTNO, DNAME, LOC) VALUES(20, 'RESEARCH', 'DALLAS');
INSERT INTO DEPT01 (DEPTNO, DNAME, LOC) VALUES(30, 'SALES', 'CHICAGO');
INSERT INTO DEPT01 (DEPTNO, DNAME, LOC) VALUES(40, 'OPERATIONS', 'BOSTON');
INSERT INTO DEPT01 (DEPTNO, DNAME, LOC) VALUES(50);

COMMIT;

SELECT *
FROM DEPT01;

```



DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

• UNIQUE와 NULL의 관계

UNIQUE는 NULL 값을 예외로 간주한다. 만약 NULL 값마저도 입력되지 않게 제한 하려면 테이블 생성시 **EMPNO NUMBER(4) NOT NULL UNIQUE**처럼 두 가지 제약 조건을 기술해야 한다.

③ 데이터 디렉터리

데이터베이스 자원을 효율적으로 관리하기 위한 다양한 정보를 저장하는 시스템 테이블을 데이터 디렉터리라고 한다. 데이터 디렉터리는 사용자가 테이블을 생성하거나 사용자를 변경하는 등의 작업을 할 때 데이터베이스 서버에 의해 자동으로 갱신되는 테이블로 사용자는 데이터 디렉터리의 내용을 직접 수정하거나 삭제 할 수 없다. 이러한 데이터 디렉터리를 사용자가 조회해 보면 시스템이 직접 관리하는 테이블이기에 암호 같은 기호만 보여질 뿐 내용을 알 수 없다. 데이터 디렉터리 원 테이블은 직접 조회하기란 거의 불가능한 일이다.

의미 있는 자료 조회가 불가능하기에 오라클은 사용자가 이해할 수 있는 데이터를 산출해 줄 수 있도록 하기 위해서 데이터 디렉터리에서 파생한 데이터 디렉터리 뷰를 제공한다.

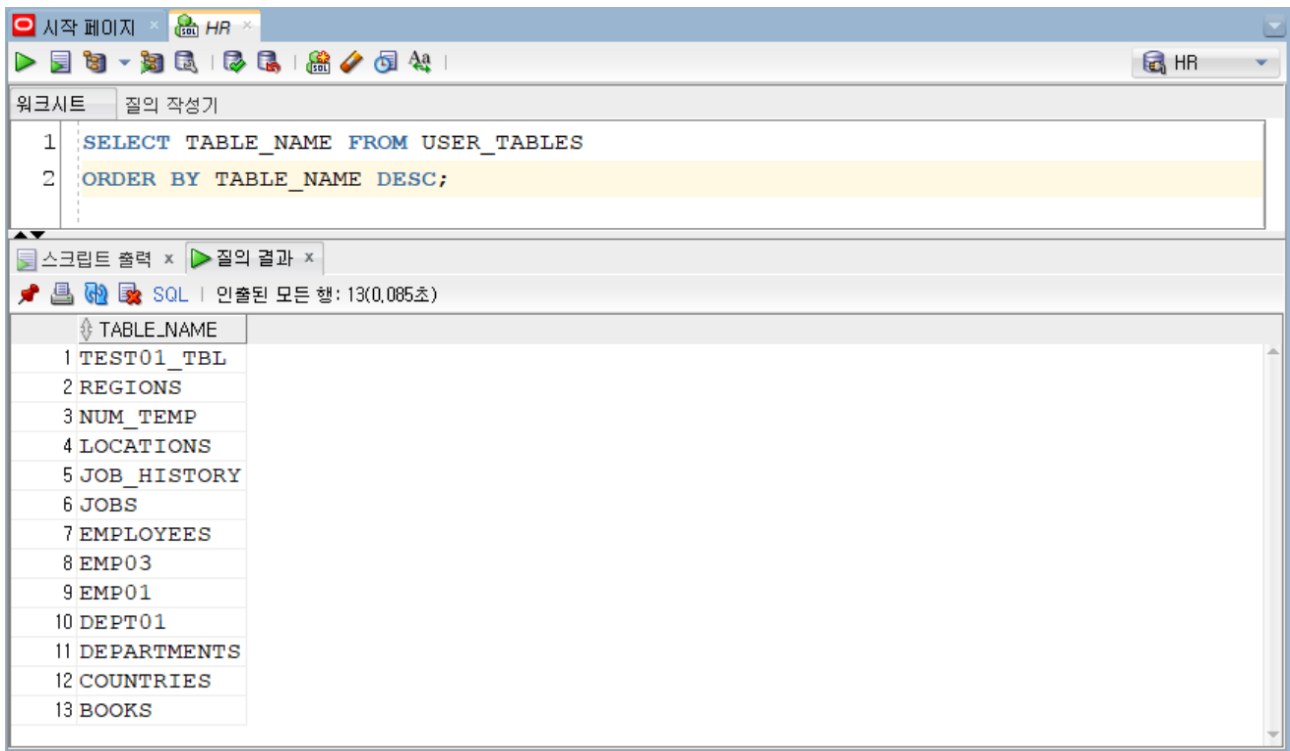
• 데이터 디렉터리 뷰는 접두어 따라 다음의 세 종류가 있다.

접두어	의 미
DBA_XXXX	데이터베이스 관리자만 접근 가능한 객체 등의 정보 조회

ALL_XXXX	자신 계정 소유 또는 권한을 부여 받은 객체 등에 관한 정보 조회
USER_XXXX	자신의 계정이 소유한 객체 등에 관한 정보 조회

HR 사용자가 생성한 테이블의 이름을 조회한다

```
SELECT TABLE_NAME FROM USER_TABLES
ORDER BY TABLE_NAME DESC;
```

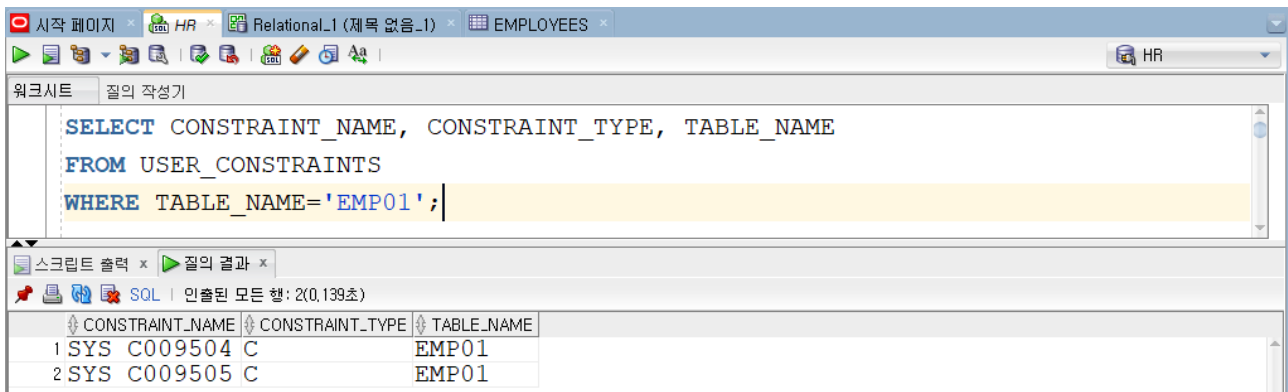


④ 제약조건 확인하기

제약조건(CONSTRAINTS)의 에러 메시지에 대한 정확한 원인을 알기 위해 오라클에서 제공해 주는 USER_CONSTRAINTS 데이터 디렉터리가 있다.

USER_CONSTRAINTS 데이터 디렉터리는 제약조건의 정보를 위해서 위해서 많은 칼럼으로 구성되어 있지만 제약조건명(CONSTRAINT_NAME), 제약조건유형(CONSTRAINT_TYPE), 제약조건이 속한 테이블명(TABLE_NAME)만을 알아본다.

```
SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, TABLE_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='EMP01';
```

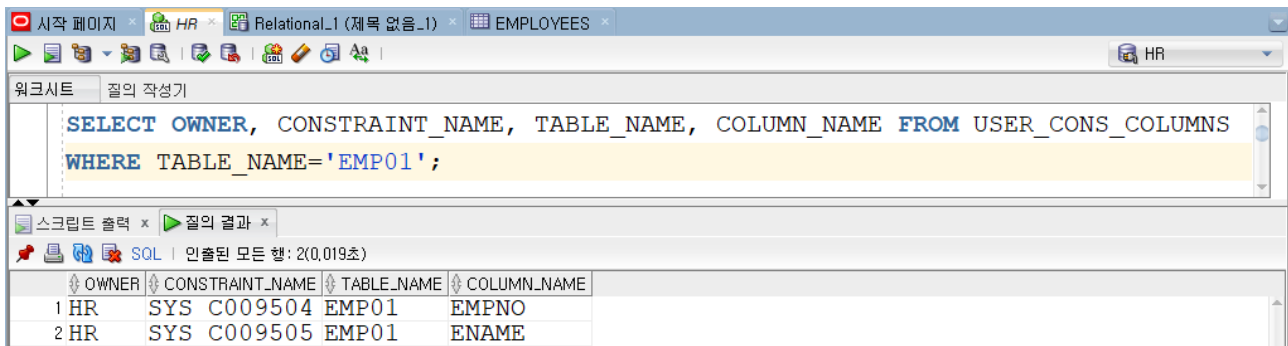



CONSTRAINT_TYPE은 P, R, U, C 4가지가 있다.

CONSTRAINT_TYPE	의 미
P	PRIMARY KEY
R	FOREIGN KEY
U	UNIQUE
C	CHECK NOT NULL

USER_CONS_COLUMNS 데이터 딕셔너리는 제약조건이 지정된 칼럼명도 저장한다.

```
SELECT OWNER, CONSTRAINT_NAME, TABLE_NAME, COLUMN_NAME FROM USER_CONS_COLUMNS
WHERE TABLE_NAME='EMP01';
```



⑤ 데이터의 구분을 위한 PRIMARY KEY 제약 조건

식별 기능을 갖는 칼럼은 유일하면서도 NULL 값을 허용하지 말아야 한다.

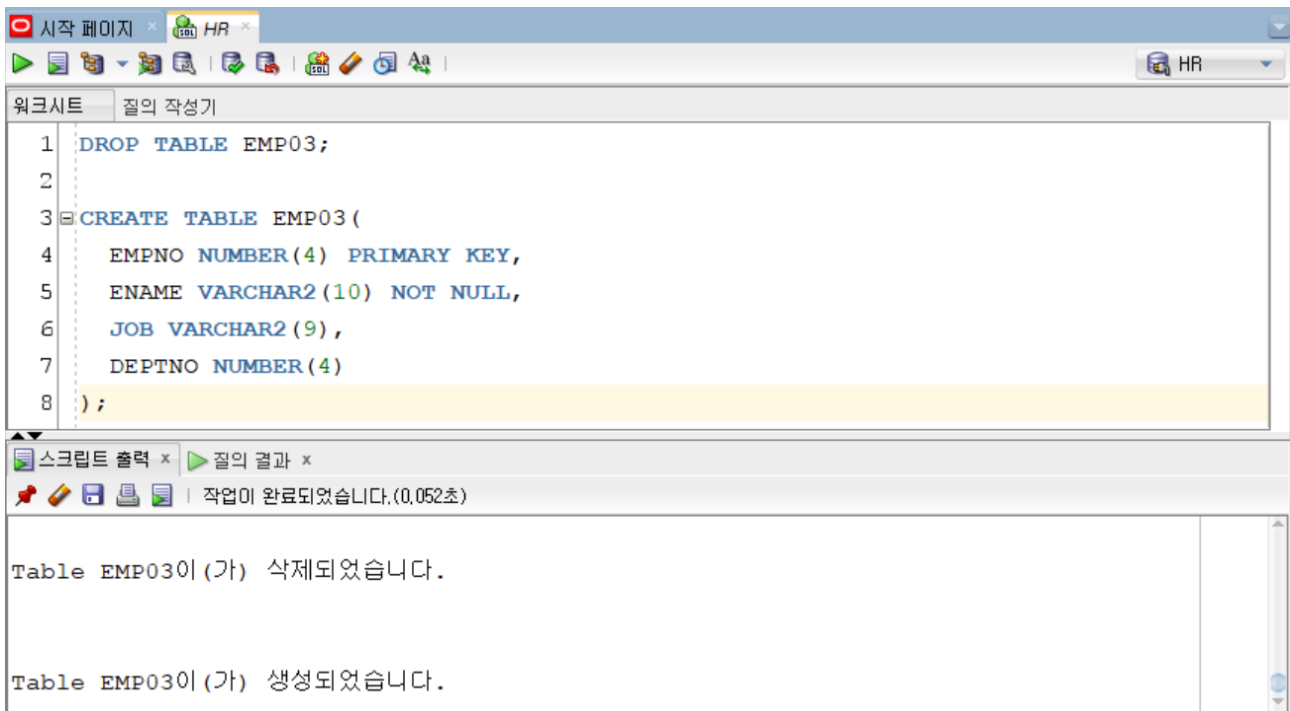
즉, UNIQUE 제약 조건과 NOT NULL 제약 조건을 모두 갖고 있어야 하는데 이러한 두 가지 제약 조건을 모두 갖는 것이 기본 키(PRIMARY KEY) 제약 조건이다.

사원 테이블의 사원번호를 기본키로 지정

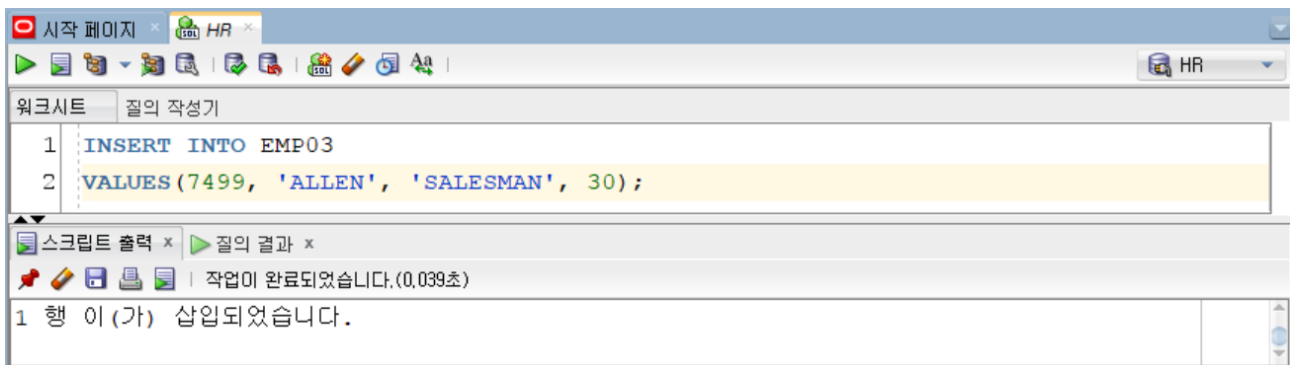
```
DROP TABLE EMP03;
```

```
CREATE TABLE EMP03(
  EMPNO NUMBER(4) PRIMARY KEY,
  ENAME VARCHAR2(10) NOT NULL,
```

```
JOB VARCHAR2(9),
DEPTNO NUMBER(4)
);
```



```
INSERT INTO EMP03
VALUES(7499, 'ALLEN', 'SALESMAN', 30);
```

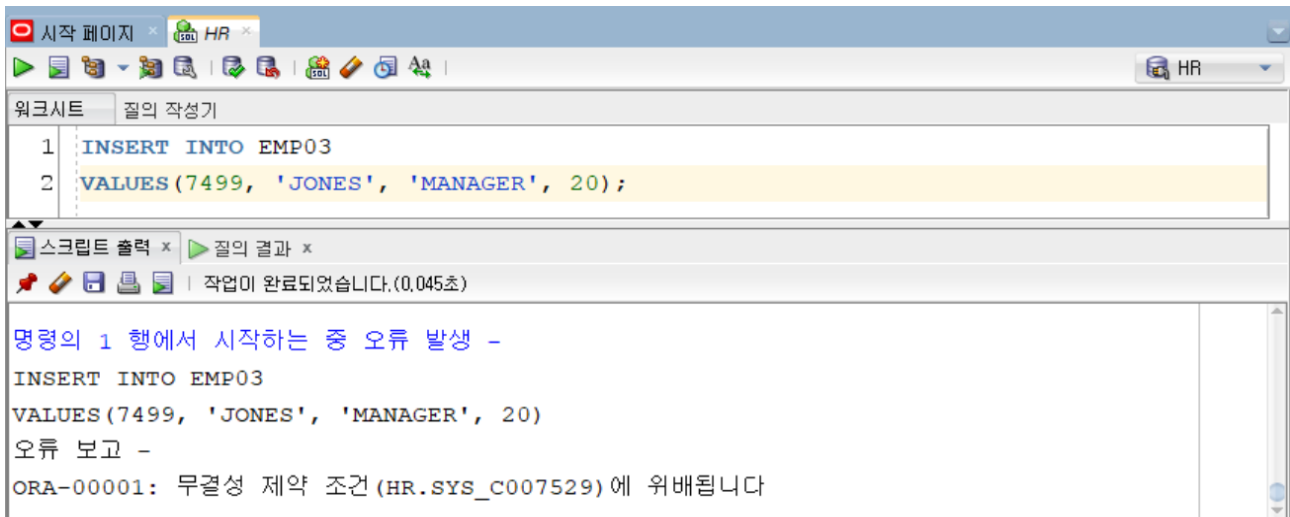


```
SELECT *
FROM EMP03;
```

	EMPNO	ENAME	JOB	DEPTNO
1	7499	ALLEN	SALESMAN	30

동일한 사원 번호를 입력하면 무결성 제약조건 위배 에러가 발생한다.

```
INSERT INTO EMP03
VALUES(7499, 'JONES', 'MANAGER', 20);
```

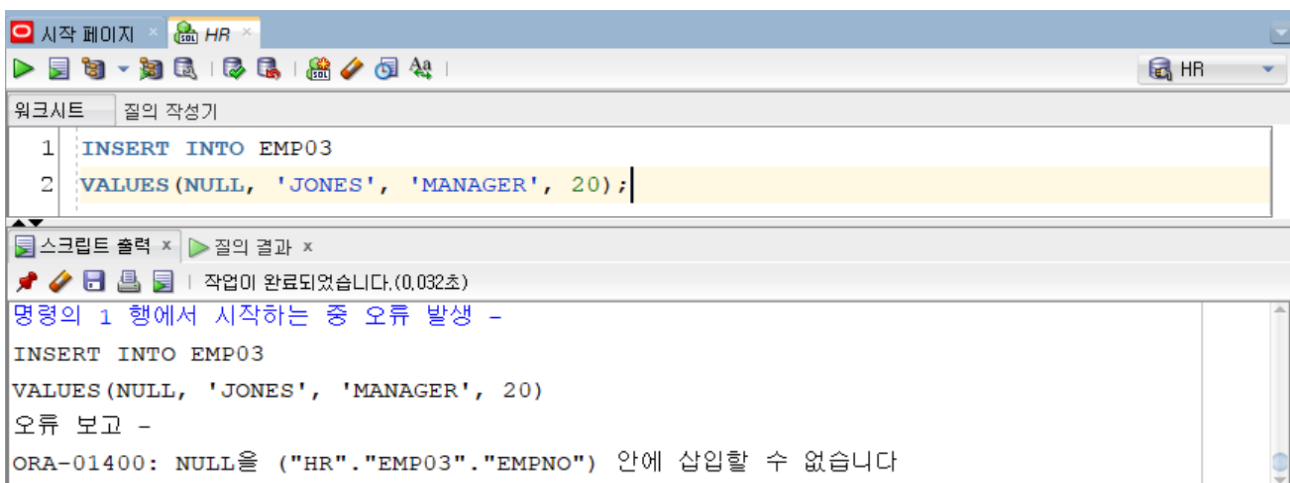


사원 번호에 NULL로 지정하고 저장해도 오류가 발생한다.

```

INSERT INTO EMP03
VALUES(NULL, 'JONES', 'MANAGER', 20);

```



```

SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, TABLE_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='EMP03';

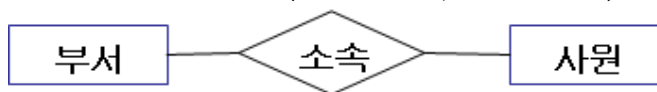
```

시작 페이지

HR

⑥ 참조 무결성을 위한 FOREIGN KEY 제약 조건

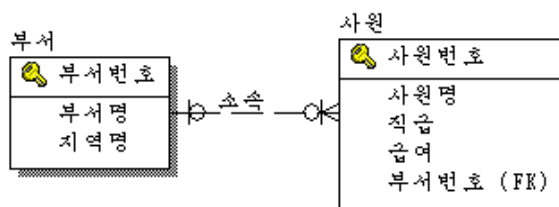
참조의 무결성은 두 테이블 사이(사원 테이블, 부서 테이블)의 주종 관계에서 설정된다.



먼저 존재해야 하는 테이블이 주체가 되는 테이블이므로 부서 테이블이 부모 테이블이 되고, 이를 참조하는 테이블인 사원 테이블이 자식 테이블이 된다.

소속이란 관계는 두 테이블 간의 참조 무결성이란 개념을 포함한 외래 키 제약 조건을 명시해야만 설정된다.

외래 키 제약 조건은 자식 테이블인 사원 테이블의 부서번호(DEPTNO) 칼럼에 부모 테이블인 부서 테이블의 부서번호를 부모키로 설정하는 것이다.



부모키가 되기 위한 칼럼은 반드시 부모 테이블의 기본 키나 유일키로 설정되어 있어야 한다.

외래 키 제약 조건에 지정하지 않은 EMP03 테이블에 부서 테이블에 존재하지 않은 50번 부서번호를 저장해 보도록 하자.

The image is a screenshot of a web-based SQL IDE interface. At the top, the SQL statement `INSERT INTO EMP03 VALUES(7566, 'JONES', 'MANAGER', 50);` is displayed. Below the statement is a toolbar with various icons for file operations and editing. The main workspace is divided into two tabs: '워크시트' (Worksheet) and '질의 작성기' (Query Editor). The '질의 작성기' tab is active, showing the SQL statement with line numbers 1 and 2. Below the workspace is another toolbar with icons for script execution and viewing results. The '스크립트 출력' (Script Output) tab is active, displaying the message '1 행 이 (가) 삽입되었습니다.' (1 row has been inserted). The status bar at the bottom indicates '작업이 완료되었습니다.(0.031초)' (Work completed. (0.031 seconds)).

SELECT * FROM EMP03;	SELECT * FROM DEPT01;
-------------------------	--------------------------

EMPNO	ENAME	JOB	DEPTNO	소속 부서가 존재하지 않는다.	DEPTNO	DNAME	LOC
1	7499 ALLEN	SALESMAN	30		1	10 ACCOUNTING	NEW YORK
2	7566 JONES	MANAGER	50		2	20 RESEARCH	DALLAS
					3	30 SALES	CHICAGO
					4	40 OPERATIONS	BOSTON

외래키 제약 조건은 EMP04 테이블을 생성시 칼럼명과 자료형을 기술한 후에 REFERENCES를 기술하면 된다. DEPTNO 칼럼을 참조하게 외래키 제약조건을 설정한다.

```
CREATE TABLE EMP04(
  EMPNO NUMBER(4) PRIMARY KEY,
  ENAME VARCHAR2(10) NOT NULL,
  JOB VARCHAR2(9),
  DEPTNO NUMBER(4) REFERENCES DEPT01(DEPTNO)
);
```

The screenshot shows the SQL Developer interface. The 'SQL Worksheet' tab contains the following SQL code:

```
1 CREATE TABLE EMP04 (
2   EMPNO NUMBER(4) PRIMARY KEY,
3   ENAME VARCHAR2(10) NOT NULL,
4   JOB VARCHAR2(9),
5   DEPTNO NUMBER(4) REFERENCES DEPT01(DEPTNO)
6 );
```

The 'Script Output' tab shows the message: "Table EMP04이 (가) 생성되었습니다." (Table EMP04 has been created).

```
INSERT INTO EMP04
VALUES(7499, 'ALLEN', 'SALESMAN', 30);
```

The screenshot shows the SQL Developer interface. The 'SQL Worksheet' tab contains the following SQL code:

```
1 INSERT INTO EMP04
2 VALUES (7499, 'ALLEN', 'SALESMAN', 30);
```

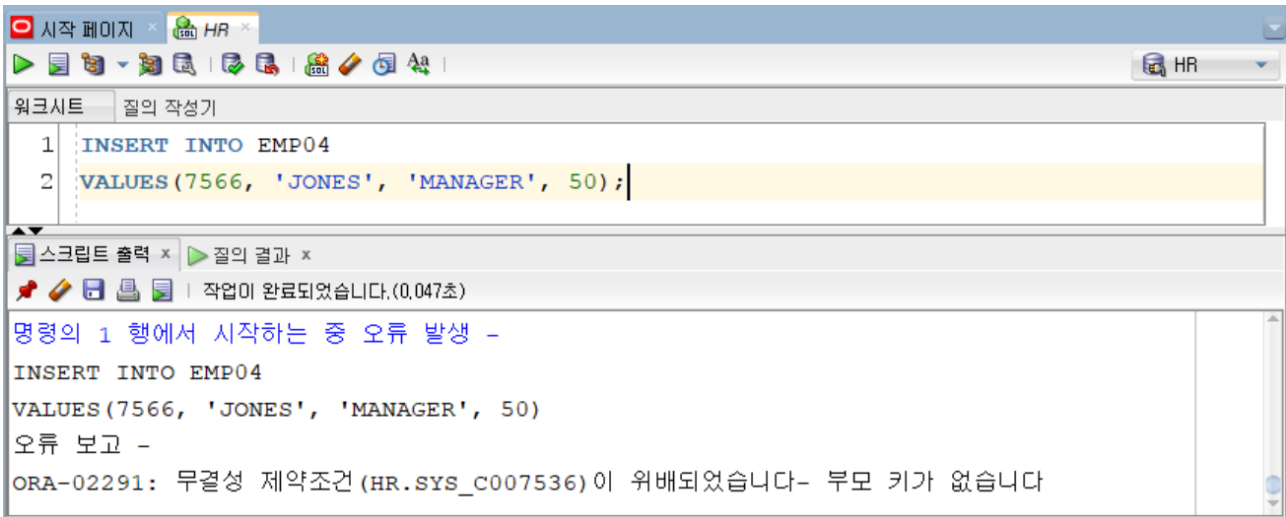
The 'Script Output' tab shows the message: "1 행 이 (가) 삽입되었습니다." (1 row has been inserted).

```
SELECT *
FROM EMP04;
```

	EMPNO	ENAME	JOB	DEPTNO
1	7499	ALLEN	SALESMAN	30

```
INSERT INTO EMP04
VALUES(7566, 'JONES', 'MANAGER', 50);
```

50번이 존재하지 않기 때문에 사원 정보가 추가되지 못하고 오류가 발생

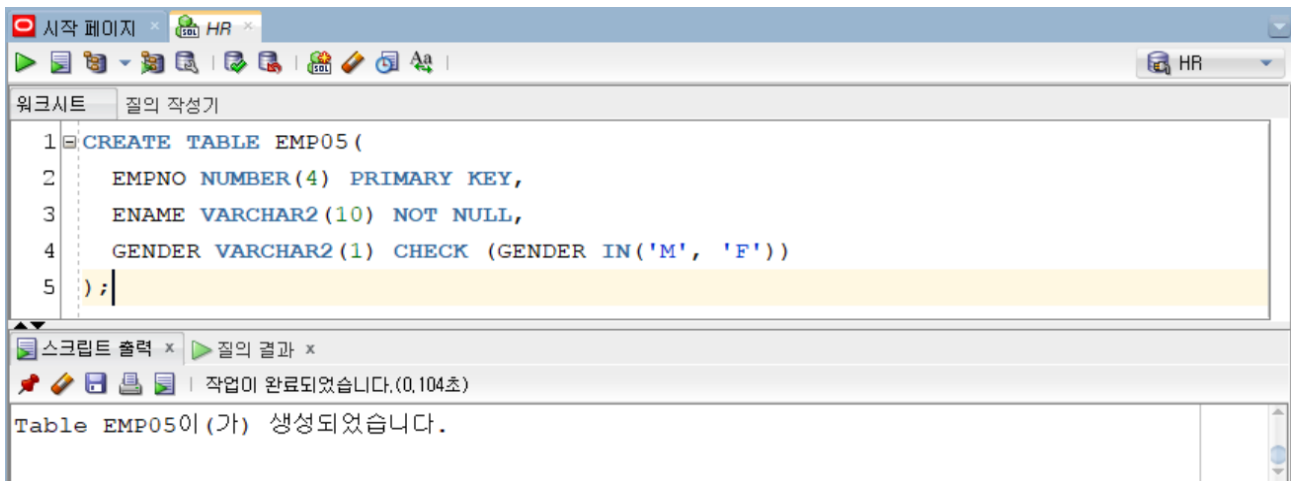


⑦ CHECK 제약 조건

CHECK 제약 조건은 입력되는 값을 체크하여 설정된 값 이외의 값이 들어오면 오류 메시지와 함께 명령이 수행되지 못하게 하는 것이다.

EMP05 사원 테이블에 GENDER(성별) 칼럼을 추가하되 GENDER칼럼에는 'M' 또는 'F'의 두 값만 저장할 수 있는 CHECK 제약조건을 설정한다.

```
CREATE TABLE EMP05(
  EMPNO NUMBER(4) PRIMARY KEY,
  ENAME VARCHAR2(10) NOT NULL,
  GENDER VARCHAR2(1) CHECK (GENDER IN('M', 'F'))
);
```



```

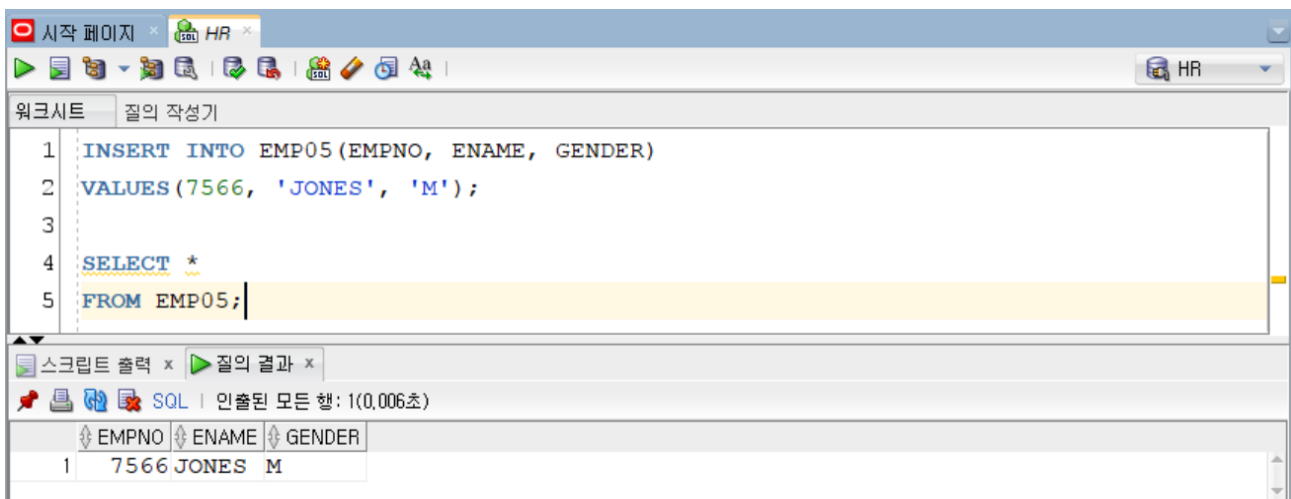
INSERT INTO EMP05(EMPNO, ENAME, GENDER)
VALUES(7566, 'JONES', 'M');

```

```

SELECT *
FROM EMP05;

```



```

INSERT INTO EMP05(EMPNO, ENAME, GENDER)
VALUES(7566, 'JONES', 'A');

```



EMP06 테이블을 새롭게 생성한다. (칼럼 레벨 형식)

```

CREATE TABLE EMP06(
  EMPNO NUMBER(4) PRIMARY KEY,

```

```

ENAME VARCHAR2(10) NOT NULL,
JOB VARCHAR2(9) UNIQUE,
DEPTNO NUMBER(4) REFERENCES DEPT01(DEPTNO)
);

```

2) 제약 조건(칼럼 레벨 형식)

EMP06 테이블을 새롭게 생성한다. (칼럼 레벨 형식)

```

CREATE TABLE EMP06(
  EMPNO NUMBER(4) PRIMARY KEY,
  ENAME VARCHAR2(10) NOT NULL,
  JOB VARCHAR2(9) UNIQUE,
  DEPTNO NUMBER(4) REFERENCES DEPT01(DEPTNO)
);

```

```

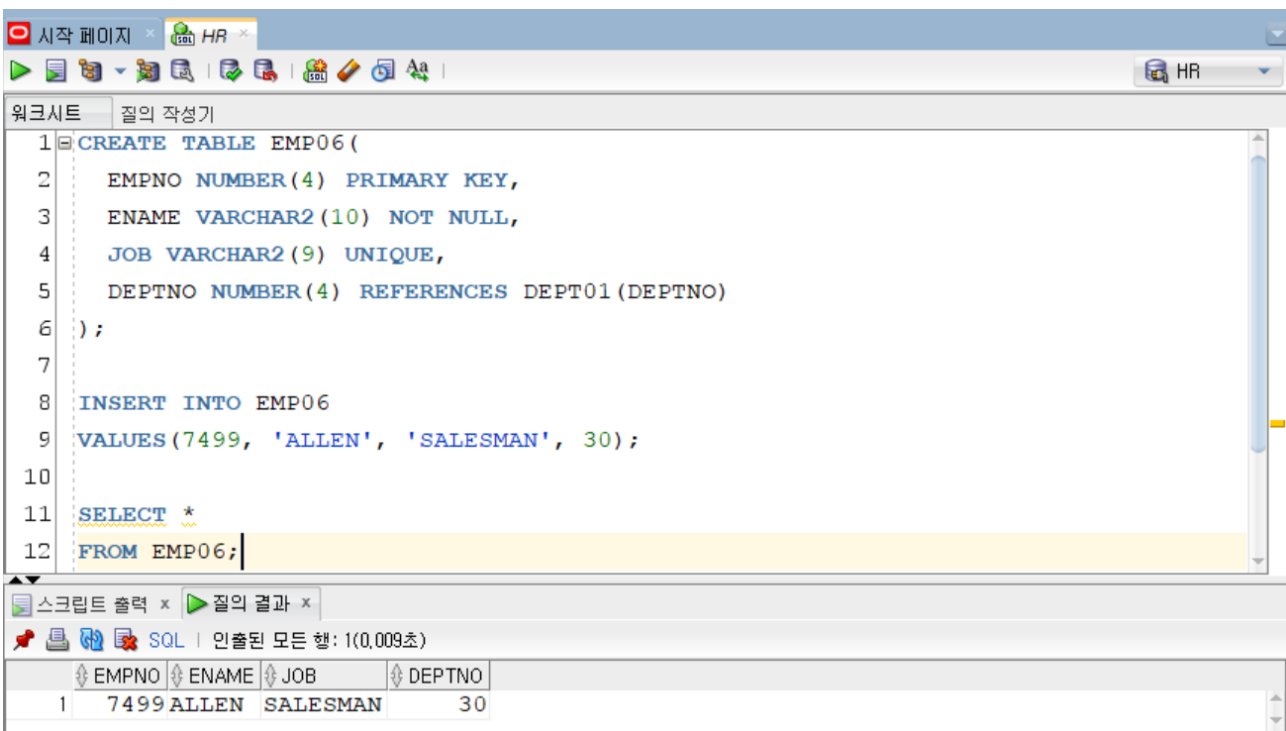
INSERT INTO EMP06
VALUES(7499, 'ALLEN', 'SALESMAN', 30);

```

```

SELECT * FROM EMP06;

```



다음은 어떤 제약조건에 위배되는지 확인해 보자.

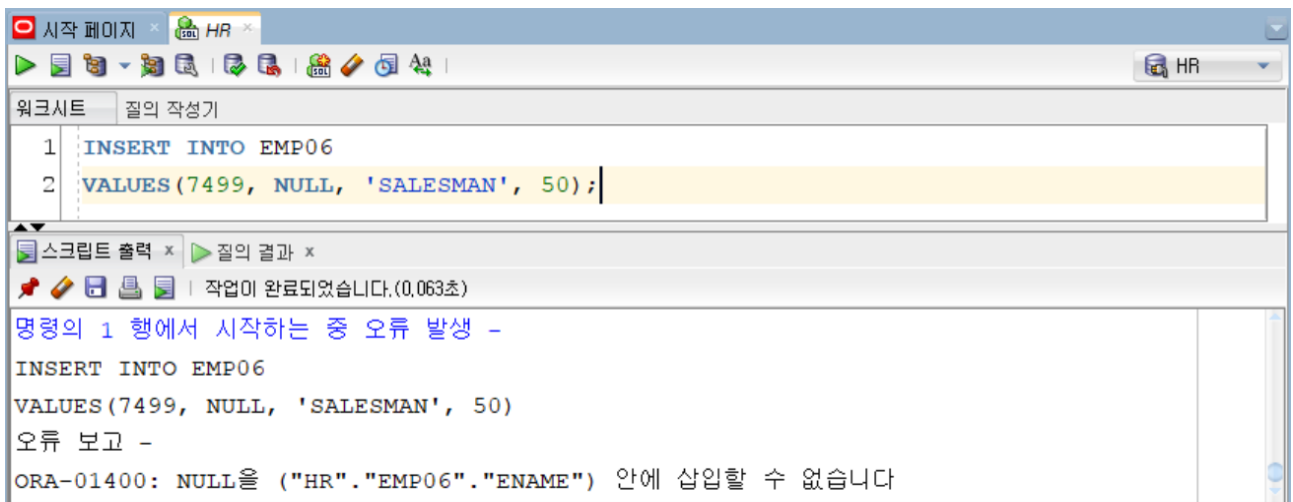
```

INSERT INTO EMP06
VALUES(7499, 'ALLEN', 'SALESMAN', 30);

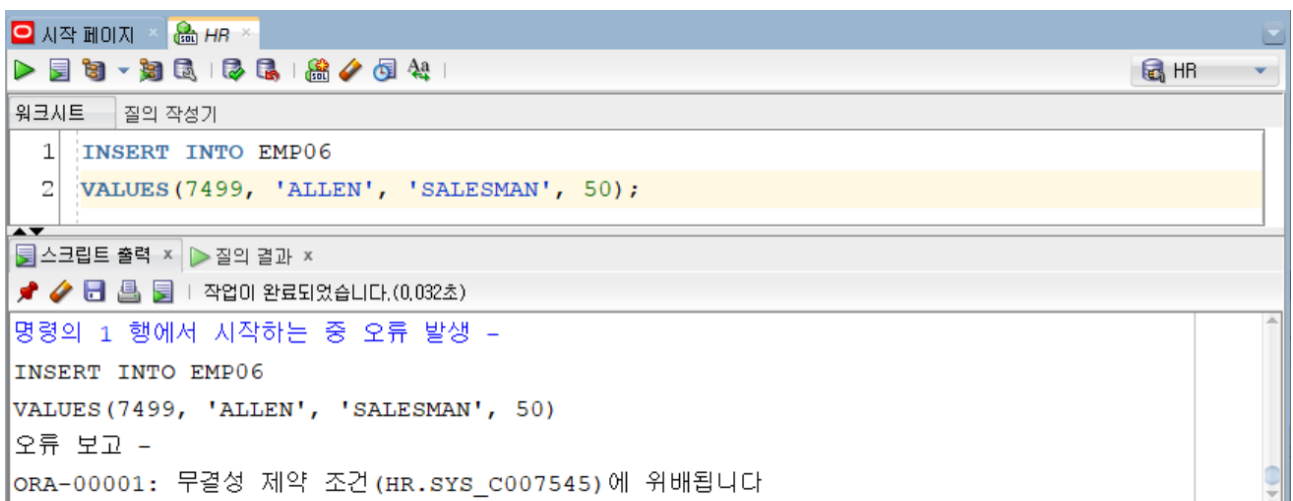
```



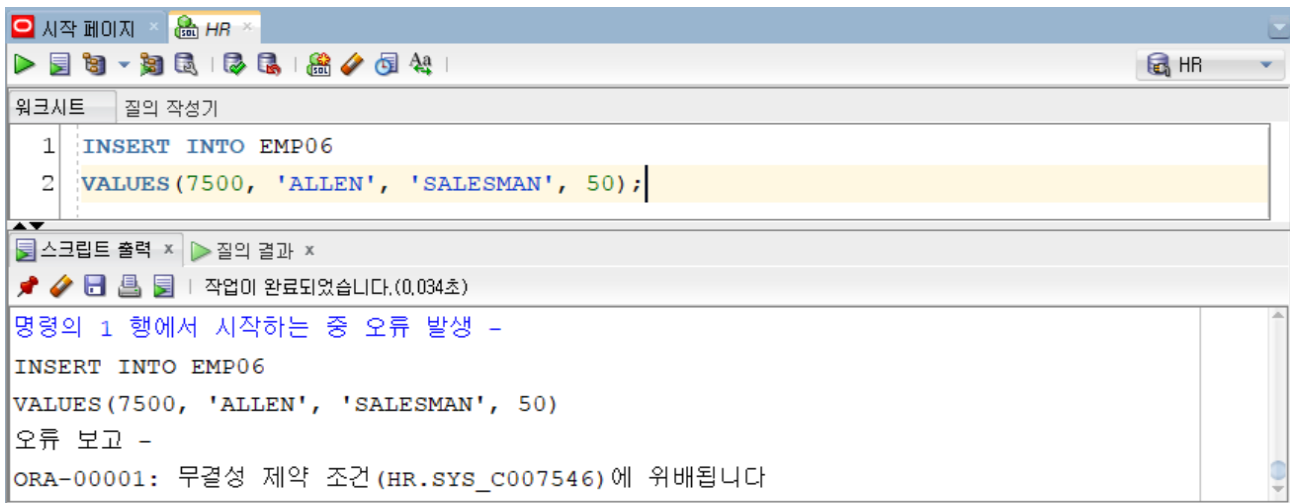

```
INSERT INTO EMP06
VALUES(7499, NULL, 'SALESMAN', 50);
```



```
INSERT INTO EMP06
VALUES(7499, 'ALLEN', 'SALESMAN', 50);
```



```
INSERT INTO EMP06
VALUES(7500, 'ALLEN', 'SALESMAN', 50);
```



3) 테이블 레벨 방식으로 제약 조건 지정하기

- 복합키로 기본키를 지정할 경우

복합키 형태로 제약조건을 지정할 경우에는 컬럼 레벨 형식으로는 불가능하고 반드시 테이블 레벨 방식을 사용해야 한다. 테이블 레벨의 제약조건 지정은 컬럼을 모두 정의하고 나서 테이블 정의를 마무리 짓기 전에 따로 생성된 컬럼들에 대한 제약조건을 한꺼번에 지정하는 것이다.

- ALTER TABLE로 제약 조건을 추가할 때

테이블의 정의가 완료되어서 이미 테이블의 구조가 결정된 후에 나중에 테이블에 제약 조건을 추가하고자 할 때에는 테이블 레벨 방식으로 제약 조건을 지정해야 한다.

- 테이블 레벨 정의 방식의 기본 형식

```
CREATE TABLE table_name (
    column_name1 datatype1,
    column_name2 datatype2,
    ...
    [CONSTRAINT constraint_name] constraint_type (column_name)
)
```

- 컬럼 레벨로 제약조건을 지정하는 방식

```
CREATE TABLE EMP07(
    EMPNO NUMBER(4) PRIMARY KEY,
    ENAME VARCHAR2(10) NOT NULL,
    JOB VARCHAR2(9) UNIQUE,
    DEPTNO NUMBER(4) REFERENCES DEPT(DEPTNO)
```

```
);
```

- 테이블 레벨로 제약조건을 지정하는 방식

```
CREATE TABLE EMP08(  
    EMPNO NUMBER(4),  
    ENAME VARCHAR2(10) NOT NULL,  
    JOB VARCHAR2(9),  
    DEPTNO NUMBER(4),  
    PRIMARY KEY(EMPNO),  
    UNIQUE(JOB),  
    FOREIGN KEY(DEPTNO) REFERENCES DEPT(DEPTNO)  
);
```

4) 제약 조건 변경하기

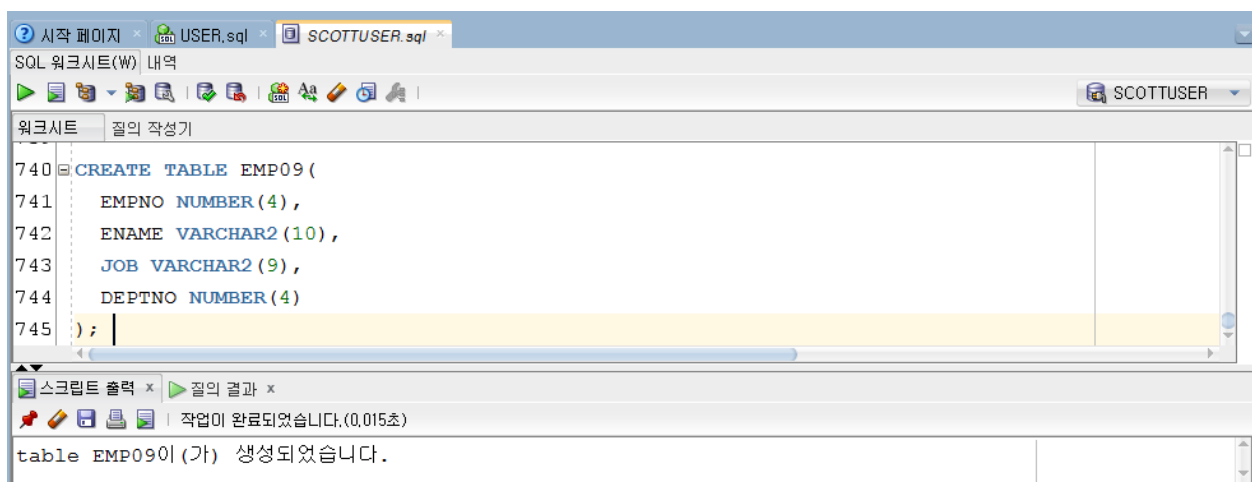
① 제약조건 추가하기

테이블 생성이 끝난 후에 제약 조건을 추가하기 위해서는 ALTER TABLE로 추가해 주어야 한다.

```
ALTER TABLE table_name  
ADD [CONSTRAINT constraint_name] constraint_type (column_name);
```

아무런 제약 조건도 지정하지 않고 EMP09 테이블을 생성한다.

```
CREATE TABLE EMP09(  
    EMPNO NUMBER(4),  
    ENAME VARCHAR2(10),  
    JOB VARCHAR2(9),  
    DEPTNO NUMBER(4)  
);
```



EMP09 테이블의 EMPNO칼럼에 기본키를 설정하고 DEPTNO칼럼에 외래키를 설정한다.

```
ALTER TABLE EMP09
```

```
ADD PRIMARY KEY(EMPNO);
```

```
ALTER TABLE EMP09
```

```
ADD CONSTRAINT EMP09_DEPTNO_FK FOREIGN KEY(DEPTNO) REFERENCES DEPT01(DEPTNO);
```

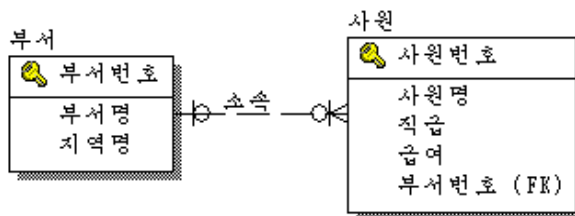
```
table EMP09이 (가) 변경되었습니다.  
table EMP09이 (가) 변경되었습니다.
```

5) 제약 조건의 비활성화와 CASCADE

제약조건의 비활성화란 설정된 제약조건을 잠시 사용하지 않게 하는 것이다.

- **DISABLE CONSTRAINT** : 제약조건의 일시 비활성화
- **ENABLE CONSTRAINT** : 비활성화된 제약 조건을 해제하여 다시 활성화

제약조건을 비활성화 하기 위해 2개의 테이블을 생성한다.



```
DROP TABLE EMP01;  
DROP TABLE EMP03;  
DROP TABLE EMP04;  
DROP TABLE EMP05;  
DROP TABLE EMP06;  
DROP TABLE DEPT01;
```

```
CREATE TABLE DEPT01(  
    DEPTNO NUMBER(2),  
    DNAME VARCHAR2(14),  
    LOC VARCHAR2(13),  
    CONSTRAINT DEPT01_DEPTNO_PK PRIMARY KEY(DEPTNO)  
);
```

워크시트 | 질의 작성기

```

1 DROP TABLE EMP01;
2 DROP TABLE EMP03;
3 DROP TABLE EMP04;
4 DROP TABLE EMP05;
5 DROP TABLE EMP06;
6 DROP TABLE DEPT01;
7
8 CREATE TABLE DEPT01 (
9     DEPTNO NUMBER(2),
10    DNAME VARCHAR2(14),
11    LOC VARCHAR2(13),
12    CONSTRAINT DEPT01_DEPTNO_PK PRIMARY KEY (DEPTNO)
13 );

```

스크립트 출력 x | 질의 결과 x

작업이 완료되었습니다.(0.051초)

Table DEPT01이 (가) 생성되었습니다.

샘플데이터를 추가

```

INSERT INTO DEPT01
VALUES(10, 'ACCOUNTING', 'NEW YORK');
INSERT INTO DEPT01
VALUES(20, 'RESEARCH', 'DALLAS');

SELECT *
FROM DEPT01;

```

시작 페이지 | HR

워크시트 | 질의 작성기

```

1 INSERT INTO DEPT01
2 VALUES(10, 'ACCOUNTING', 'NEW YORK');
3 INSERT INTO DEPT01
4 VALUES(20, 'RESEARCH', 'DALLAS');
5
6 SELECT *
7 FROM DEPT01;

```

스크립트 출력 x | 질의 결과 x

SQL | 인출된 모든 행: 2(0.007초)

DEPTNO	DNAME	LOC
1	10 ACCOUNTING	NEW YORK
2	20 RESEARCH	DALLAS

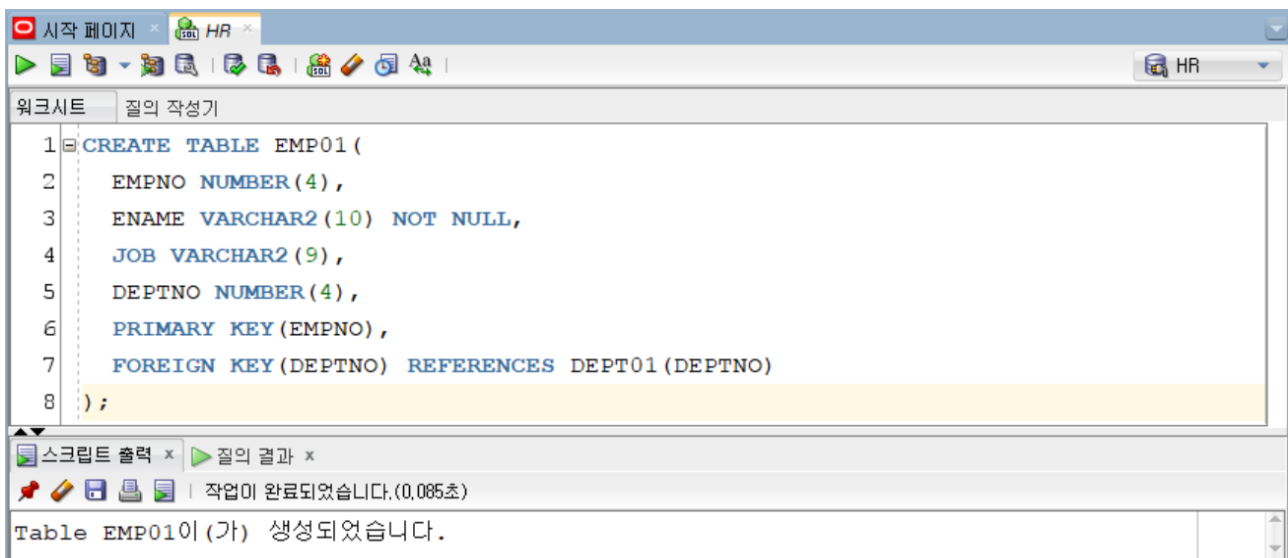
사원 테이블의 부서 번호가 부서 테이블의 부서 번호를 참조 할 수 있도록 외래키를 설정한다.

```
CREATE TABLE EMP01(
```

```

EMPNO NUMBER(4),
ENAME VARCHAR2(10) NOT NULL,
JOB VARCHAR2(9),
DEPTNO NUMBER(4),
PRIMARY KEY(EMPNO),
FOREIGN KEY(DEPTNO) REFERENCES DEPT01(DEPTNO)
);

```



샘플데이터를 추가

```

INSERT INTO EMP01
VALUES(7499, 'ALLEN', 'SALESMAN', 10);
INSERT INTO EMP01
VALUES(7369, 'SMITH', 'CLERK', 20);

SELECT *
FROM EMP01;

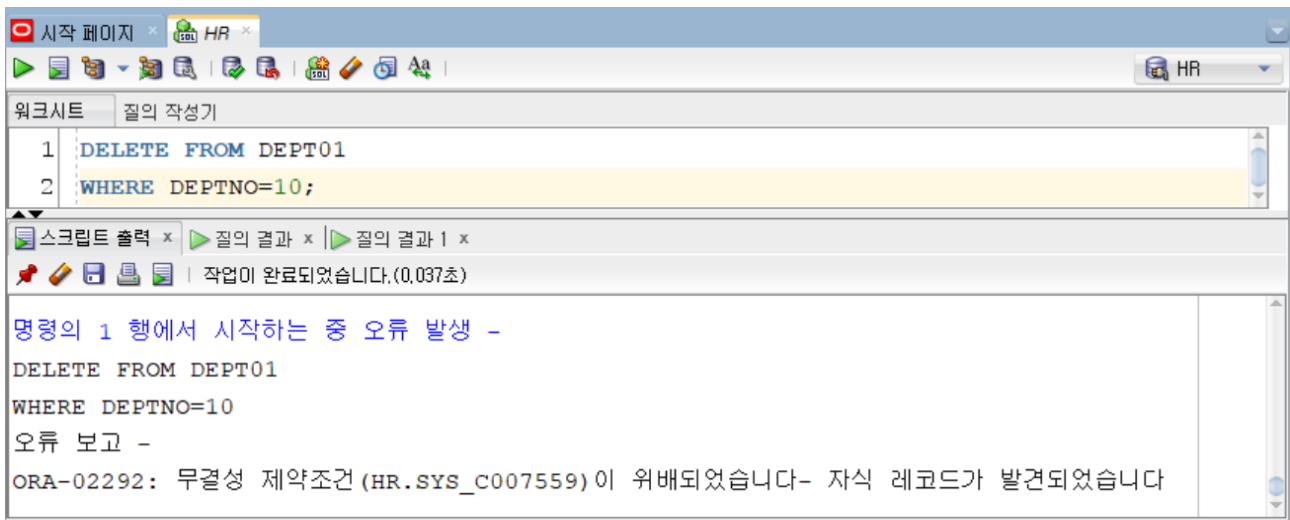
```

	EMPNO	ENAME	JOB	DEPTNO
1	7499	ALLEN	SALESMAN	10
2	7369	SMITH	CLERK	20

```

DELETE FROM DEPT01
WHERE DEPTNO=10;

```



자식테이블인 EMP01은 부모테이블인 DEPT01의 기본키인 부서번호를 참조하고 있어 삭제할 수 없다.

부서번호가 10번인 자료가 삭제되도록 하기 위해서는 아래와 같이 해야한다.

- 부서 테이블(EMP01)의 10번 부서에서 근무하는 사원을 삭제한 후 부서 테이블(DEPT01)에서 10번 부서를 삭제한다.
- 참조 무결성 때문에 삭제가 불가능하므로 EMP01 테이블의 외래키 제약 조건을 제거한 후에 10번 부서를 삭제한다.

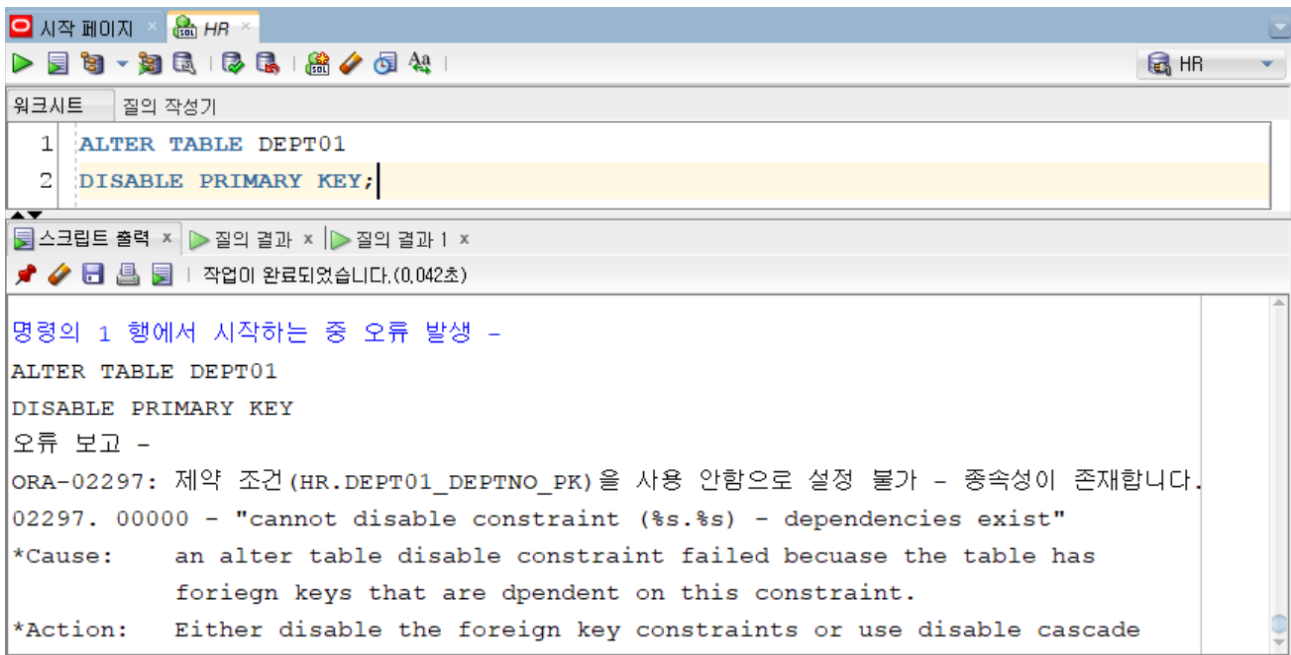
① CASCADE 옵션

CASCADE 옵션은 부모 테이블과 자식 테이블 간의 참조 설정이 되어 있을 때 부모 테이블의 제약 조건을 비활성화하면 이를 참조하고 있는 자식 테이블의 제약조건까지 같이 비활성화 시켜주는 옵션이다.

DEPT01 테이블의 기본키 제약조건을 비활성화 한다.

제약조건을 모르더라도 'PRIMARY KEY'로 비활성화할 수 있다. 그러나 DEPT01 테이블의 기본키는 EMP01 테이블의 외래키에서 참조하고 있기 때문에 제약조건을 비활성화 할 수 없다.

```
ALTER TABLE DEPT01
DISABLE PRIMARY KEY;
```

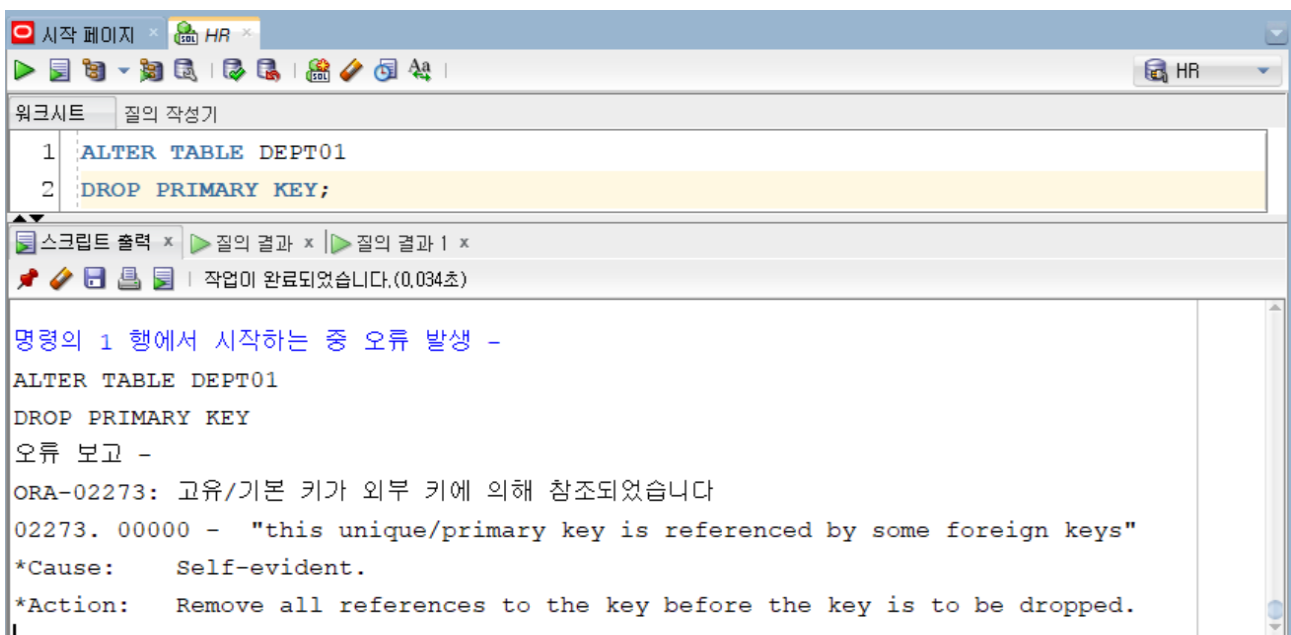


- 부모테이블의 기본키에 대한 제약 조건을 비활성화하기 위한 작업을 순서대로 정리해보면
 부모테이블의 기본키를 참조하는 자식 테이블의 외래키에 대한 제약 조건을 비활성화해야 한다.
 부모 테이블의 기본키에 대한 제약조건을 비활성화해야 한다.

```
ALTER TABLE DEPT01
DISABLE PRIMARY KEY CASCADE;
```

DEPT01 테이블의 기본키 제약조건을 삭제해 보며 오류발생

```
ALTER TABLE DEPT01
DROP PRIMARY KEY;
```



CASCADE 옵션을 지정하여 기본키 제약 조건을 삭제하게 되면 이를 참조하는 외래키 제약조건도

연속적으로 삭제된다.

```
ALTER TABLE DEPT01  
DROP PRIMARY KEY CASCADE;
```

[예제]

1. 학과테이블을 생성한다. (subject)

- 일련번호(no) / 학과번호(s_num) / 학과명(s_name)
- 일련번호는 중복을 허용하지 않는 컬럼으로 설정한다.
- 모든 컬럼은 반드시 입력되어야 한다.
- 학과테이블에서 학과번호는 2자리로 구성한다.
(01-컴퓨터학과 / 02-교육학과 / 03-신문방송학과 / 04-인터넷비즈니스과 / 05-기술경영과)

2. 학생테이블을 생성한다. (student)

- 일련번호(no) / 학번(sd_num) / 이름(sd_name)/ 아이디(sd_id) / 비밀번호(sd_passwd) / 학과번호(s_num) / 생년월일(sd_birth) / 핸드폰번호(sd_phone) / 주소(sd_address) / 이메일(sd_email) / 등록일자(sd_date)
- 일련번호는 중복을 허용하지 않는 컬럼으로 설정한다.
- 모든 컬럼은 반드시 입력되어야 한다.
- 학생테이블에서 학번은 8자리로 구성한다. (연도2자리+학과2자리+일련번호 - 예로06010001)

학번	이름	아이디	주소
06010001	김정수	javajsp	서울시 서대문구 창전동
95010002	김수현	jdbcmmania	서울시 서초구 양재동
98040001	공지영	gonji	부산광역시 해운대구 반송동
02050001	조수영	water	대전광역시 중구 은행동
94040002	최경란	novel	경기도 수원시 장안구 이목동
08020001	안익태	korea	본인의 주소

위에 제시되지 않은 나머지 데이터는 임의값으로 입력한다.

3. 과목테이블을 생성한다. (lesson)

- 일련번호(no)/ 과목약어(l_abbre) / 과목명(l_name) /
- 일련번호는 중복을 허용하지 않는 컬럼으로 설정한다.
- 모든 컬럼은 반드시 입력되어야 한다.
- 과목테이블에서 과목약어(abbreviation)와 과목명은 [K-국어 / M-수학 / E-영어 / H-역사 / P-프로그래밍 / D-데이터베이스 / ED-교육학이론]로 이루어진다.

4. 수강테이블을 생성한다. (trainee)

- 일련번호(no)/학번(sd_num)/과목약어(l_abbre)/ 과목구분(t_section) / 등록일자(t_date)
- 모든 컬럼은 반드시 입력되어야 한다.
- 수강테이블에서 과목구분은 culture(교양) / major(전공) / minor(부전공)로 이루어진다.
(교양 - 국어 / 수학 / 영어 / 역사, 전공 - 컴퓨터, 데이터베이스)

5. 제약사항

- 테이블에 존재하는 등록일자는 기본값으로 오늘날짜로 입력되도록 설정한다.
- 학과 테이블에 있는 학과번호를 학생테이블에 있는 학과번호만 입력되어야 한다.
- 수강테이블에는 학번은 반드시 학생테이블에 있는 학번만 입력되어야 하고,
과목약어는 반드시 과목테이블에 있는 과목약어만 입력되어야 한다.