# 10. 테이블 생성과 관리

컴퓨터공학부

김은경

# [학습 목표]

1. DB 객체의 종류에 따른 특성을 나열할 수 있다.
2. 테이블 생성 원칙에 맞추어 새로운 테이블을 생성할 수 있다.
3. 테이블 생성 시 다른 테이블의 데이터를 복사할 수 있다.
4. ALTER TABLE 명령을 이용하여 기존 테이블의 구조를 변경할 수 있다.
5. DROP TABLE 명령이나 TRUNCATE TABLE 명령으로 테이블을 삭제할 수 있다.

# 1. DB 객체

## DB 객체(object)란?

- 논리적인 데이터 저장 영역 구조로서, 스키마(Schema) 객체라고도 칭함

## DB 객체의 종류

| 함수 종류 | 특징 |
|---|---|
| 테이블<br>(Table) | 행과 열로 구성된 기본적인 데이터 저장 영역의 기본 단위 |
| 뷰<br>(View) | 하나 이상의 테이블로부터 획득한 논리적인 관점에서의<br>데이터의 부분 집합 |
| 시퀀스<br>(Sequence) | 자동 생성되는 일련 번호로, 대개 기본 키 값 생성에 사용됨 |
| 인덱스<br>(Index) | SQL 문의 실행 속도를 향상시키기 위해 생성하는 색인 |
| 동의어<br>(Synonym) | DB 객체에 부여한 별칭 |

# 1. DB 객체

## 객체 이름 지정 규칙

① 첫 문자 : 반드시 영문자로 시작. 단, 한글 이름은 예외

② 길이 : 이름의 길이는 1~30 사이

③ 사용 문자 : A-Z, a-z, 0-9, _(underscore), $, #(비권장), 한글

④ 이름 중복 여부 : 동일한 사용자가 소유한 객체 이름은 중복될 수 없음

⑤ 예약어 사용 여부 : 객체 이름으로 예약어를 사용할 수 없음

⑥ 대소문자 구분 여부 : 대소문자를 구분하지 않음
(자동으로 대문자로 변경됨)

⑦ 대소문자 구분 방법 : 대소문자를 구분하고 싶으면, 이중 인용부호로 묶어서 지정함

# 1. DB 객체

## 객체 이름 변경 방법

```
RENAME   old_name   TO   new_name ;
```

① 인덱스를 제외한, 테이블, 뷰, 시퀀스, 동의어 이름의 변경이 가능함

② 객체의 소유자만 변경할 수 있음

# 2. 데이터 형

## 대표적인 데이터 형

| 데이터 형 | 특징 |
|---|---|
| **CHAR** | 고정 길이 문자를 기억함<br>(최대 2000 바이트) |
| **VARCHAR2** | 가변 길이 문자를 기억함<br>(최대 4000 바이트) |
| **VARCHAR** | VARCHAR2와 같음<br>(추후 다른 목적으로 사용될 예정임) |
| **DATE** | 날짜와 시간을 기억함<br>(기본 형식 : YY/MM/DD) |
| **NUMBER(n)** | 길이가 n인 숫자를 기억함 |
| **LONG** | 가변 길이 문자를 기억함<br>(최대 2G 바이트) |

# 3. 테이블 생성

## 테이블 생성 원칙

① 데이터 정의어(DDL) 가운데 하나인 CREATE TABLE 명령으로 생성함

② 사용자가 테이블 생성 권한을 가진 경우에만 테이블을 생성할 수 있음
(이 권한은 DB 관리자가 사용자에게 부여함)

③ 테이블 생성 후에 자동으로 커밋(COMMIT) 됨

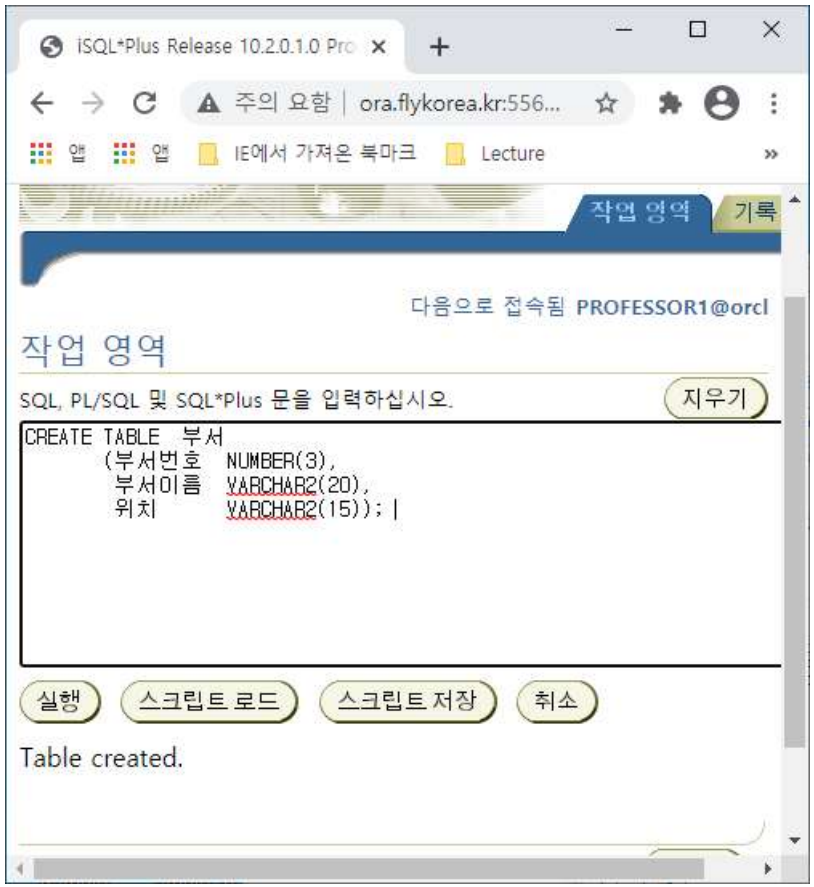
## CREATE TABLE 명령 형식

```
CREATE  TABLE  테이블이름
               (열 이름   데이터형   [DEFAULT  표현식]
               [, 열 이름   데이터형,  . . .] );
```

✓ DEFAULT 표현식 : INSERT 명령으로 행을 삽입할 때, 열의 값을 생략할 경우 열에 할당할 디폴트 값(문자열, 표현식, SQL 함수 가능)을 명시함
→ 디폴트 값을 지정함으로써 열에 NULL 값이 입력되는 것을 방지할 수 있음

# 3. 테이블 생성

## ✓ 테이블 생성 예





✓ 테이블의 구조 확인 명령어 : DESCRIBE(또는 줄여서 DESC)

# 4. 데이터 사전

## 데이터 사전(Data Dictionary)이란?

- DB 관리를 위해 자동으로 생성되는 읽기 전용 테이블의 집합
- 시스템 내에 있는 모든 객체들에 대한 정의와 명세에 관한 정보를 수록하고 있으므로, DB 관리에 있어서 매우 중요한 역할을 함
- 사용자와 시스템이 공동으로 사용함

## 사용자 객체와 관련된 데이터 사전 테이블

| 데이터 사전의 테이블 이름 | 용도 |
|---|---|
| USER_TABLES | 사용자가 소유한 모든 테이블의 이름을 알 수 있음 |
| USER_OBJECTS | 사용자가 소유한 서로 다른 객체 형을 알 수 있음 |
| USER_CATALOG (또는 CAT) | 사용자가 소유한 테이블, 뷰, 동의어 및 시퀀스를 알 수 있음 |

# 4. 데이터 사전

✓ **사용자가 소유한 테이블 이름 확인 예**

```
SELECT   TABLE_NAME
FROM     USER_TABLES;
```

| TABLE_NAME |
|---|
| NEW_EMP |
| 부서 |
| SAL_CLASS |
| PAY_RATE |
| COURSE |
| INSTRUCTOR |
| TEACHES |
| DEPT |
| EMP |
| TAKES |
| FOREIGN_EMP |

11 rows selected.

# 4. 데이터 사전

✓ **사용자가 소유한 객체 형 확인 예**

```
SELECT   DISTINCT   OBJECT_TYPE
FROM     USER_OBJECTS;
```

| OBJECT_TYPE |
| --- |
| TABLE |
| INDEX |

# 4. 데이터 사전

✓ **사용자가 소유한 테이블, 뷰, 동의어 및 시퀀스 확인 예**

```
SELECT    *
FROM      CAT;
```

| TABLE_NAME | TABLE_TYPE |
|---|---|
| NEW_EMP | TABLE |
| 부서 | TABLE |
| SAL_CLASS | TABLE |
| PAY_RATE | TABLE |
| COURSE | TABLE |
| INSTRUCTOR | TABLE |
| TEACHES | TABLE |
| DEPT | TABLE |
| EMP | TABLE |
| TAKES | TABLE |
| FOREIGN_EMP | TABLE |

11 rows selected.

✓ **사용자가 직접 생성하지 않은 테이블 등을 삭제하는 명령 : PURGE RECYCLEBIN**

# 5. 서브쿼리를 이용한 테이블 생성

## 서브쿼리를 포함하는 CREATE TABLE 명령의 형식

- 테이블 생성과 동시에 행을 삽입하기 위해서 이용함

```
CREATE  TABLE  테이블이름  [열 이름1(, 열 이름2, . . .)]
  AS    서브쿼리;
```

## 사용자 객체와 관련된 데이터 사전 테이블

① 서브쿼리는 괄호로 묶지 않음

② 열 이름 목록을 기술한 경우, 서브쿼리가 반환하는 열의 순서가 CREATE TABLE 절에 명시된 열의 순서와 반드시 일치해야 함

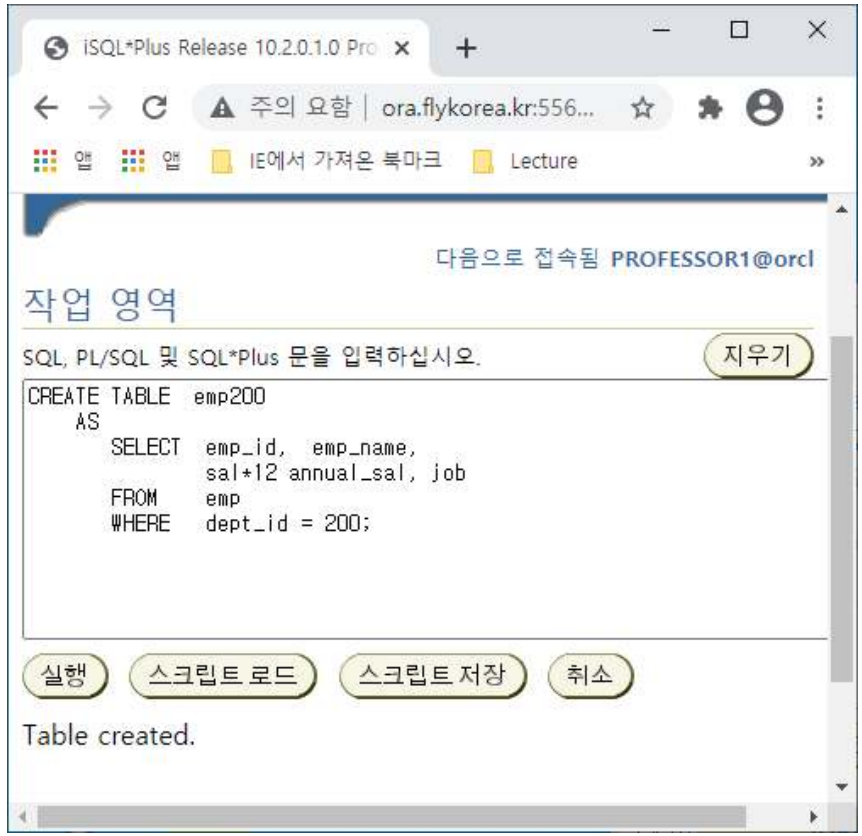
③ 열 이름 목록을 생략하면, 테이블이 생성될 때의 열의 수 및 순서가 서브쿼리가 반환한 열의 수 및 순서와 동일하게 됨

# 5. 서브쿼리를 이용한 테이블 생성

## ✓ 테이블 생성과 동시에 행을 삽입하는 예제

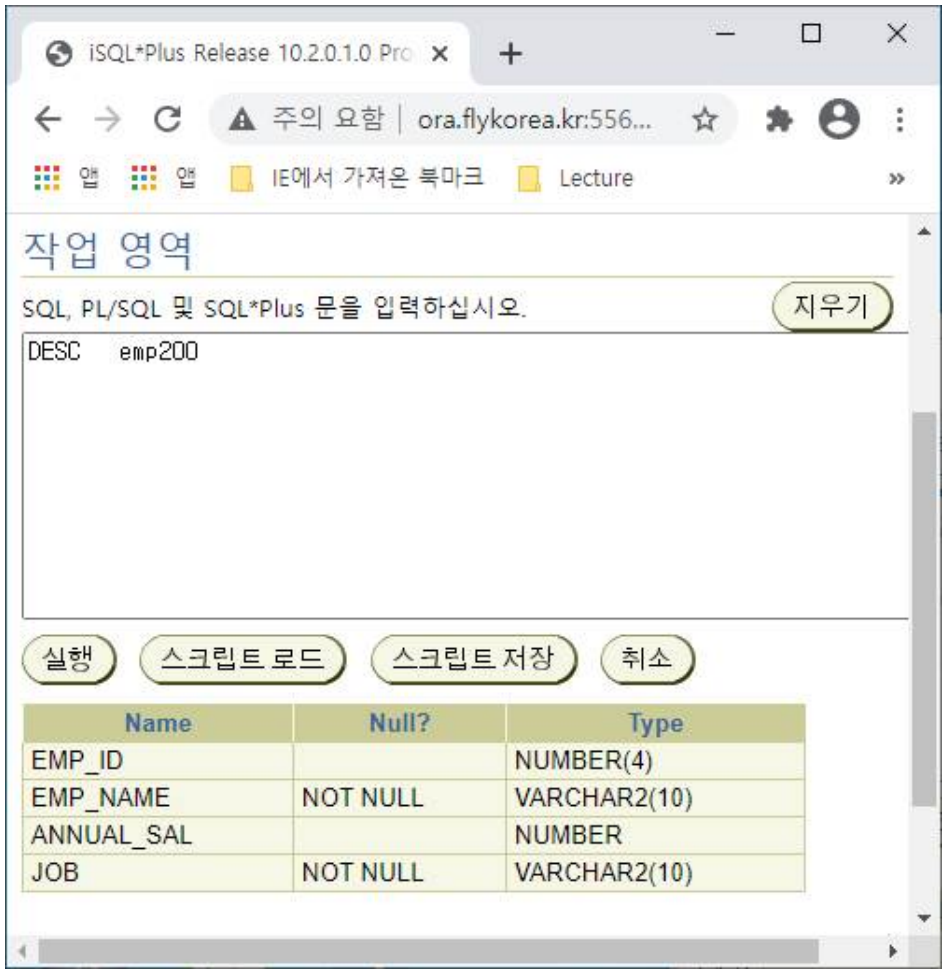
- emp 테이블에서 200번 부서의 사원 데이터 가운데 사원번호와 사원이름, 업무를 복사하고, 급여에 12를 곱해서 연봉(ANNUAL_SAL)으로 저장하는 'emp200'이라는 테이블 생성하기

```
CREATE TABLE   emp200
    AS
       SELECT   emp_id,   emp_name,
                sal*12 annual_sal, job
       FROM     emp
       WHERE    dept_id = 200;
```



# 5. 서브쿼리를 이용한 테이블 생성

→ 생성된 테이블 구조 확인 :



# 6. 테이블 구조 변경

## ALTER TABLE 명령의 형식

- 기본 테이블의 구조를 변경하기 위해서 이용함

### ① 열 추가 형식

```
ALTER  TABLE
        ADD   (열 이름1  데이터형  [DEFAULT  표현식]
              [, 열 이름2  데이터형] . . . );
```

### ② 열 변경 형식

```
ALTER  TABLE
        MODIFY   (열 이름1  데이터형  [DEFAULT  표현식]
                 [, 열 이름2  데이터형] . . . );
```

# 6. 테이블 구조 변경

## 테이블 구조 변경 허용 범위

① 새로운 열 추가

② 기존 열 변경

③ 열에 대한 디폴트 값 새로 정의

④ 테이블에서 열 삭제는 불가능함

## 테이블에 새로운 열 추가 방법

① ALTER TABLE ~ ADD 명령으로 열을 추가함

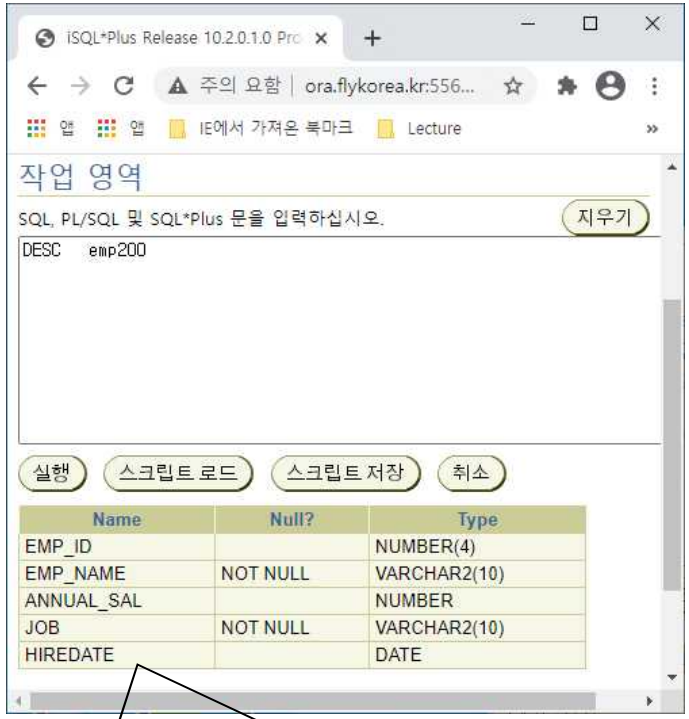
② 열의 위치는 명시할 수 없으며, 새로 추가된 열은 테이블의 마지막 열이 됨

# 6. 테이블 구조 변경

## ✓ 테이블에 열 추가하는 예제

- emp200 테이블에 DATE 형인 '입사일(hiredate)' 열 추가하기

```
ALTER TABLE  emp200
ADD   (hiredate    DATE);
```


emp200 테이블 마지막에 hiredate 라는 새로운 열이 추가됨

# 6. 테이블 구조 변경

## 기존 테이블의 열 변경 방법

① ALTER TABLE ~ MODIFY 명령으로 기존 열을 변경함

② 열의 데이터 형과 크기, 디폴트 값을 변경할 수 있음

③ 크기 확장은 항상 가능함

④ 크기 축소나 데이터 형의 변경은 데이터 값이 없는 경우에만 가능함

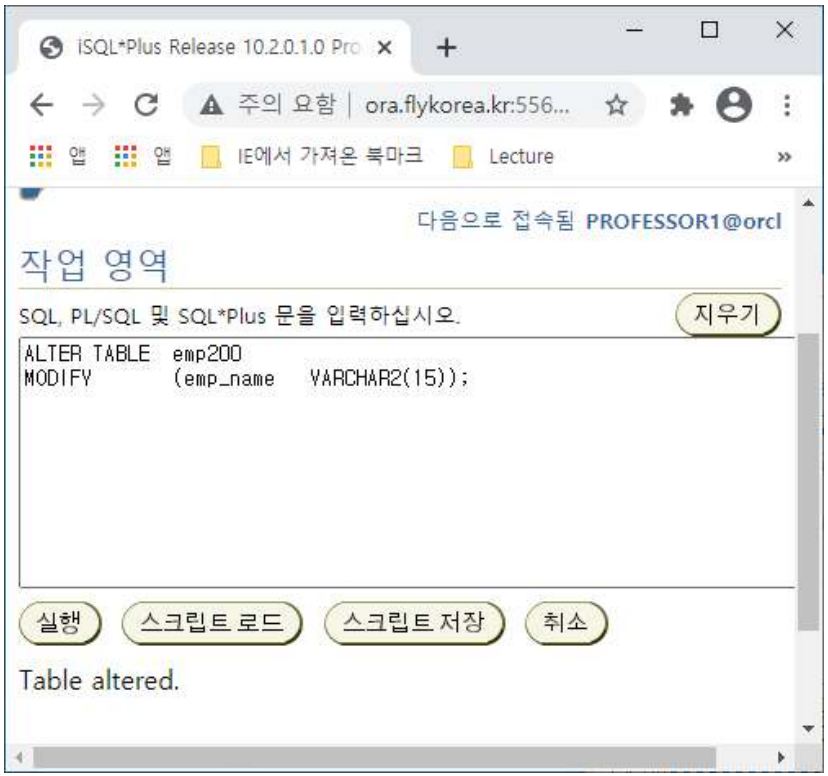
⑤ 디폴트 값을 변경한 경우, 변경된 이후에 삽입되는 데이터에만 적용됨

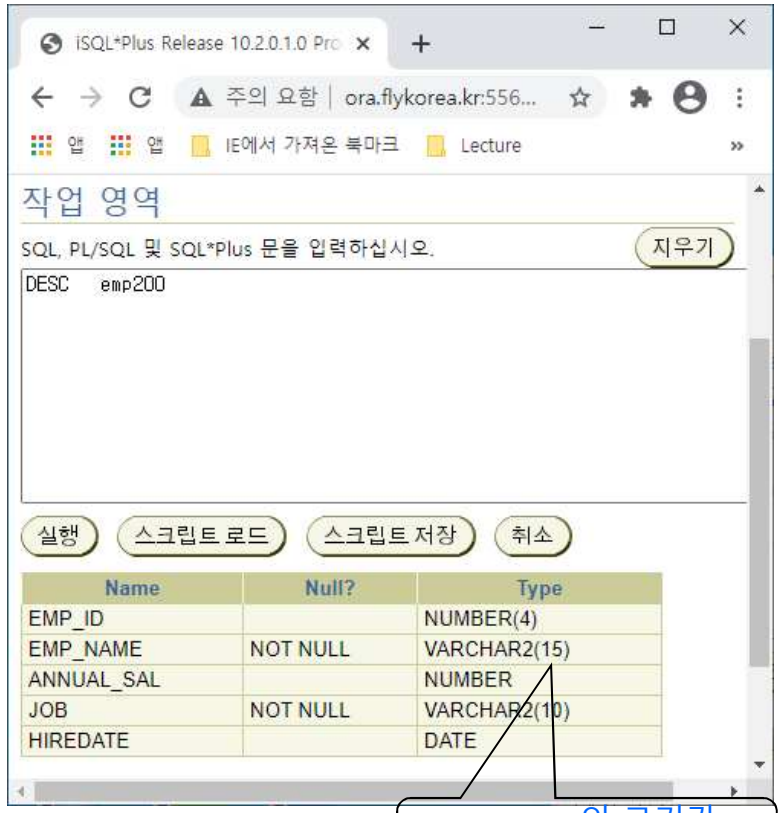
# 6. 테이블 구조 변경

## ✓ 테이블의 열 변경 예제

- emp200 테이블에서 사원이름의 크기를 15로 확장하기

```
ALTER TABLE   emp200
MODIFY        (emp_name    VARCHAR2(15));
```



# 7. 테이블 삭제

## DROP TABLE 명령의 형식

- 불필요한 테이블 구조와 함께 테이블에 저장된 데이터도 한꺼번에 삭제함

```
DROP   TABLE   테이블이름;
```

## DROP TABLE 명령으로 테이블 삭제하는 원칙

① 테이블 생성자나 권한이 부여된 사람만 삭제할 수 있음

② 테이블 구조와 함께 테이블에 저장된 모든 데이터가 삭제됨

③ 모든 미결정된 트랜잭션이 커밋(COMMIT)됨

④ 모든 인덱스가 함께 삭제됨

⑤ 삭제된 후에는 복구(ROLLBACK)가 불가능함

✓ COMMIT 명령 : 모든 미결정된 데이터 조작을 영구적으로 변경하여 현재 트랜잭션 종료함

✓ ROLLBACK 명령 : 모든 미결정된 데이터 변경을 버림으로써 현재의 트랜잭션을 종료함

# 7. 테이블 삭제

## TRUNCATE TABLE 명령의 형식

- 테이블 구조는 남겨두고 테이블에 저장된 데이터만 삭제함

```
TRUNCATE TABLE  테이블이름;
```

## TRUNCATE TABLE 명령으로 테이블 삭제하는 원칙

① 테이블 생성자나 권한이 부여된 사람만 삭제할 수 있음

② 테이블 구조는 남겨두고, 테이블에 저장된 모든 데이터만 삭제함

③ 테이블이 사용한 기억공간을 해제함

④ 모든 미결정된 트랜잭션이 자동 커밋됨

⑤ 삭제한 데이터의 복구(ROLLBACK)가 불가능함

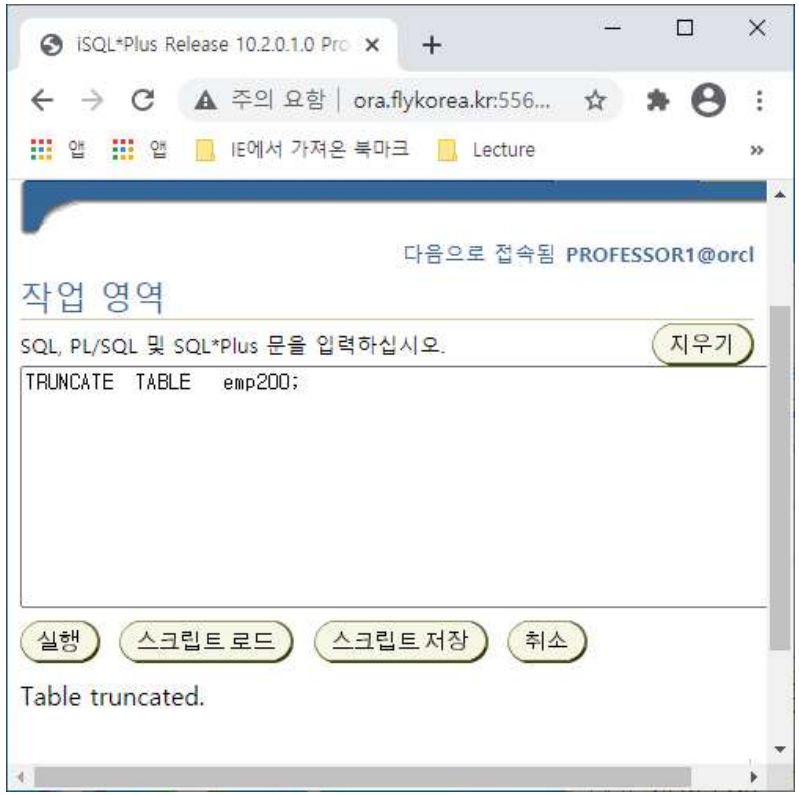
✓ 데이터 조작어인 DELETE 명령으로 데이터를 삭제하면, 기억공간을 해제하지 않게 되며, 따라서 데이터 복구(ROLLBACK)가 가능함

# 7. 테이블 삭제

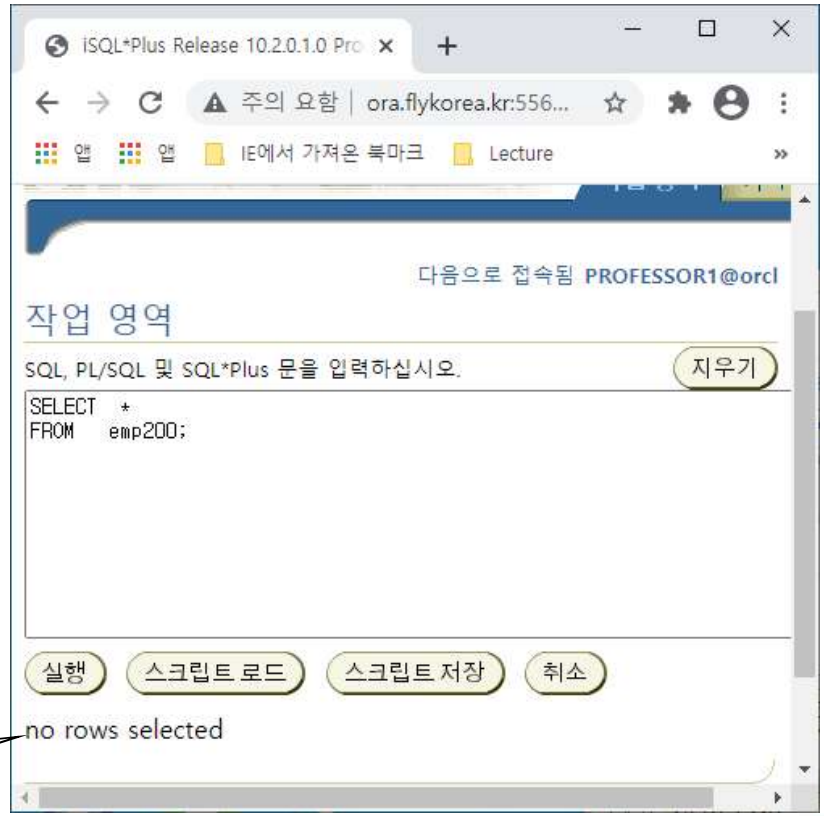
## ✓ 어떤 테이블의 데이터만 삭제하는 예제

- emp200 테이블의 모든 데이터만 삭제하기

```
TRUNCATE   TABLE    emp200;
```




```
SELECT   *
FROM    emp200;
```




Emp200 테이블의 모든 데이터가
삭제되어 선택된 행이 없음

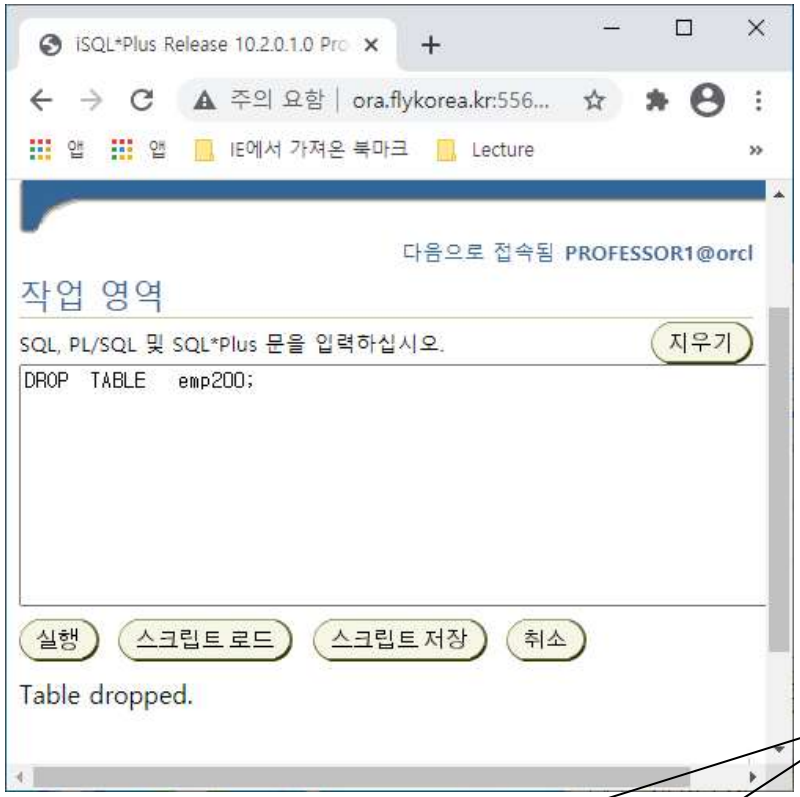
# 7. 테이블 삭제

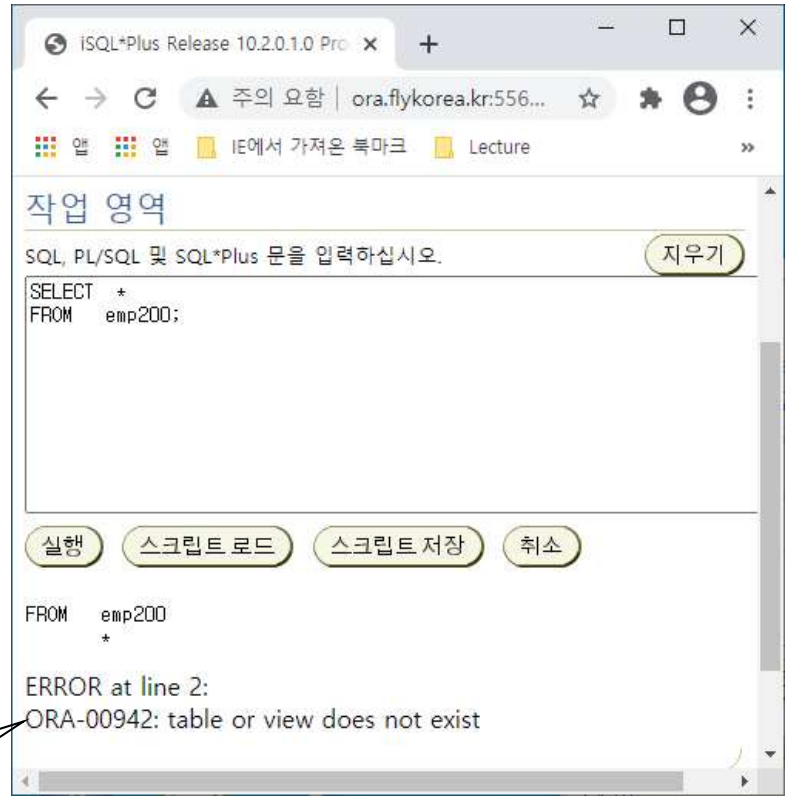
## ✓ 어떤 테이블의 구조를 삭제하는 예제

- emp200 테이블 구조 삭제하기

```
DROP   TABLE    emp200;
```

```
SELECT   *
FROM     emp200;
```




Emp200 테이블 구조가 삭제되어 emp200 테이블 자체가 존재하지 않으므로 오류가 발생함

# Q & A