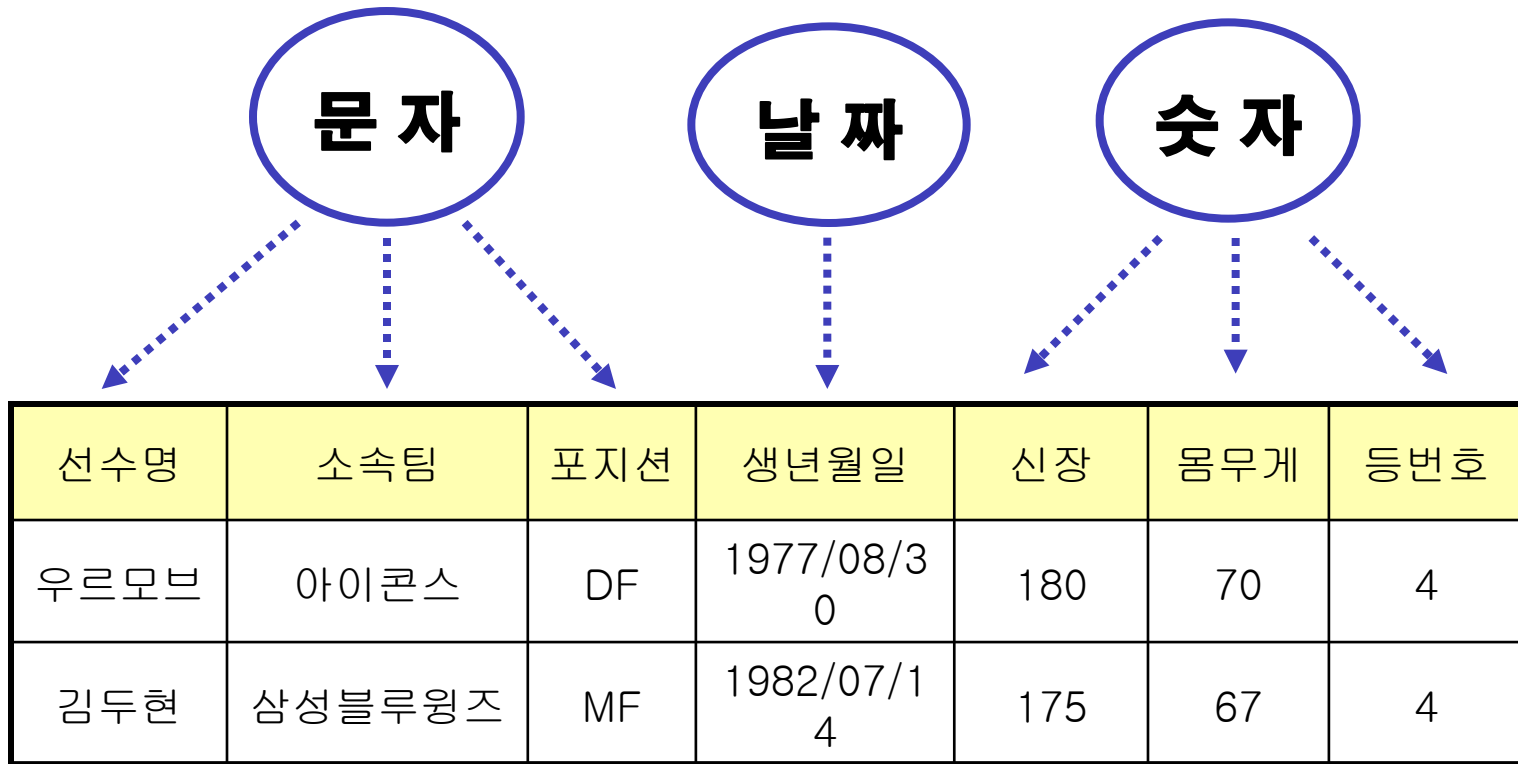


SQL 기초 정리

(수업에 필요한)

SQL 데이터



문자와 숫자를 구분하는 기준

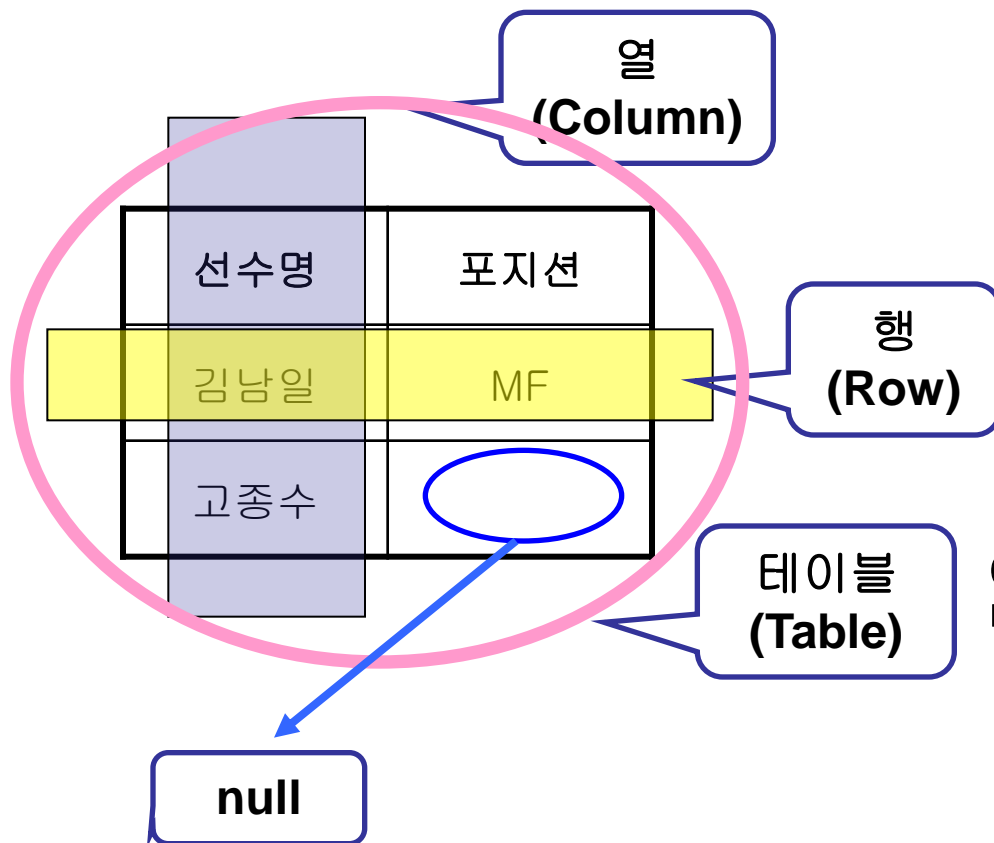
→ 산술연산 가능 여부

SQL 이란?

- **SQL**은 데이터베이스에서 데이터 조작과 데이터 정의를 위해서 사용하는 언어
- 데이터 베이스와 대화한다는 것은 데이터베이스에 자료를 입력, 수정, 삭제, 조회를 할 수 있다는 의미이다.

용 어		설 명
입력	INSERT	데이터베이스에 관리하기를 원하는 자료를 저장
수정	UPDATE	데이터베이스에 입력되어 있는 자료들을 변경
삭제	DELETE	데이터베이스에 입력되어 있는 자료들을 삭제
조회	SELECT	데이터베이스에 입력되어 있는 자료들을 보려고 할 때

테이블과 관련된 용어



**Column과 Row를 가진 2차원 구조
반드시 하나이상의 컬럼을 가진다.**



SQL 명령어와 관련된 용어

명령어의 종류	명령어
DQL (데이터 검색 조회)	SELECT
DML (데이터 조작어)	INSERT, DELETE, UPDATE
DDL (데이터 정의어)	CREATE, ALTER, RENAME DROP, TRUNCATE
TCL (트랜잭션 제어어)	COMMIT, ROLLBACK, SAVEPOINT

데이터 타입

데이터 타입	설 명
CHAR(S)	고정 길이 문자열 S : 1 - 2000
VARCHAR2(S)	가변 길이 문자열 S : 1 - 4000
NUMBER(P, S)	수치를 표현 P : 전체 숫자의 길이 S : 소수점 이하 표시
DATE	날짜와 시각을 표현

DDL - CREATE

1. 테이블 이름을 지정하고 각 칼럼들은 괄호 “()”로 묶어 지정한다.
2. 칼럼 뒤에 데이터 타입을 반드시 지정되어야 한다.
3. 각 칼럼들은 콤마 “,”로 구분되고, 항상 끝은 세미콜론 “;”으로 끝난다.
4. 한 테이블 안에서 칼럼 이름은 같을 수 없으며 다른 테이블에서의 칼럼 이름과는 같을 수 있다.

- =====
- 테이블 생성시 대.소문자 구분은 필요 없습니다.
 - **DATE** 형에는 별도의 크기를 지정하지 않습니다.
 - **VARCHAR2**형은 반드시 크기를 지정해야 합니다
 - **NUMBER, CHAR**는 크기를 정할 수 도 있고 정하지 않을 수도 있습니다.
 - 칼럼과 칼럼의 구분은 콤마(,)를 사용하며 마지막은 사용하지 않습니다.



DDL - CREATE

```
CREATE TABLE 테이블명(  
    칼럼명1 DATATYPE [DEFAULT 형식],  
    칼럼명2 DATATYPE [DEFAULT 형식],  
    . . . .  
);
```

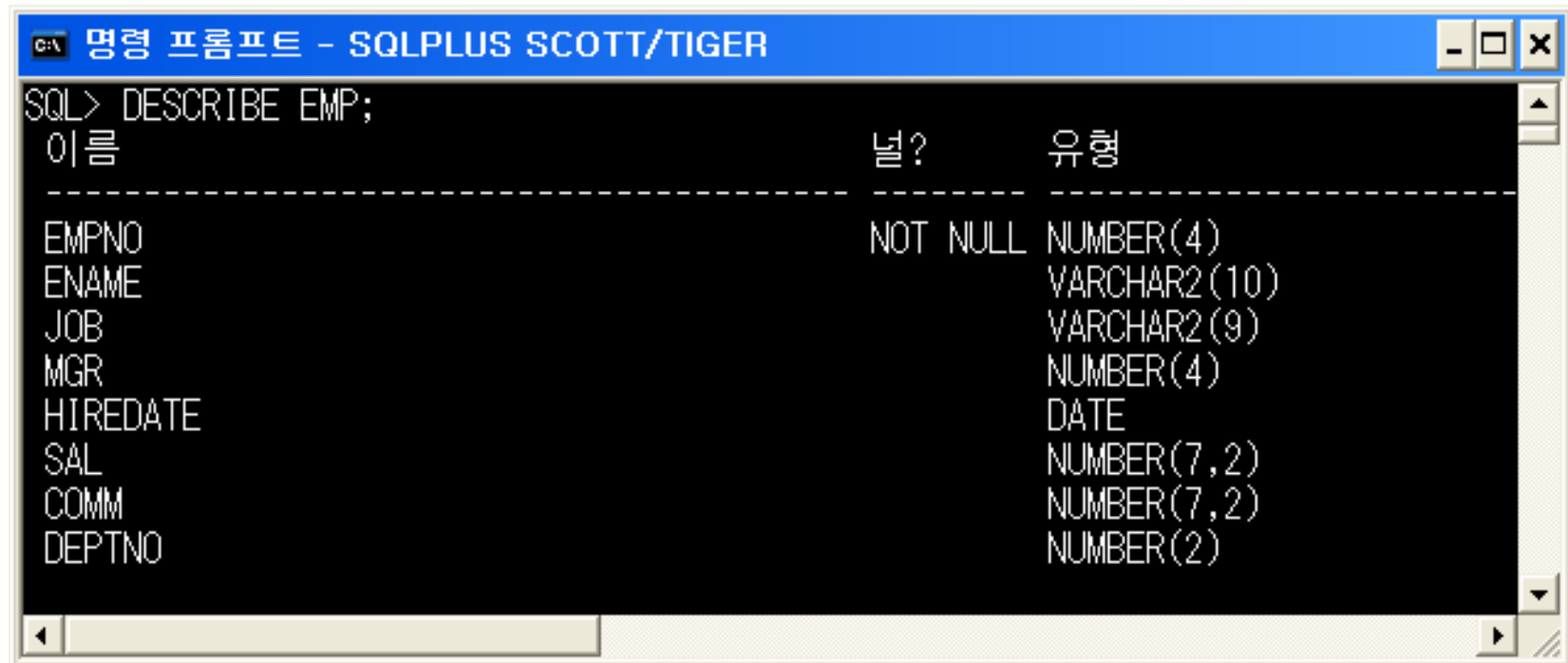
```
CREATE TABLE EX01(  
    EMPNO    VARCHAR2(10),  
    SAL      NUMBER(4) DEFAULT 0,  
    ENAME    VARCHAR2,    ➔ 오류발생  
    SEX      CHAR,        ➔ 가능  
    BIRTH    DATE         ➔ 가능  
);
```


DDL - DESCRIBE

DESCRIBE 테이블명;

DESC 테이블명;

DESCRIBE EMP; OR DESC EMP;

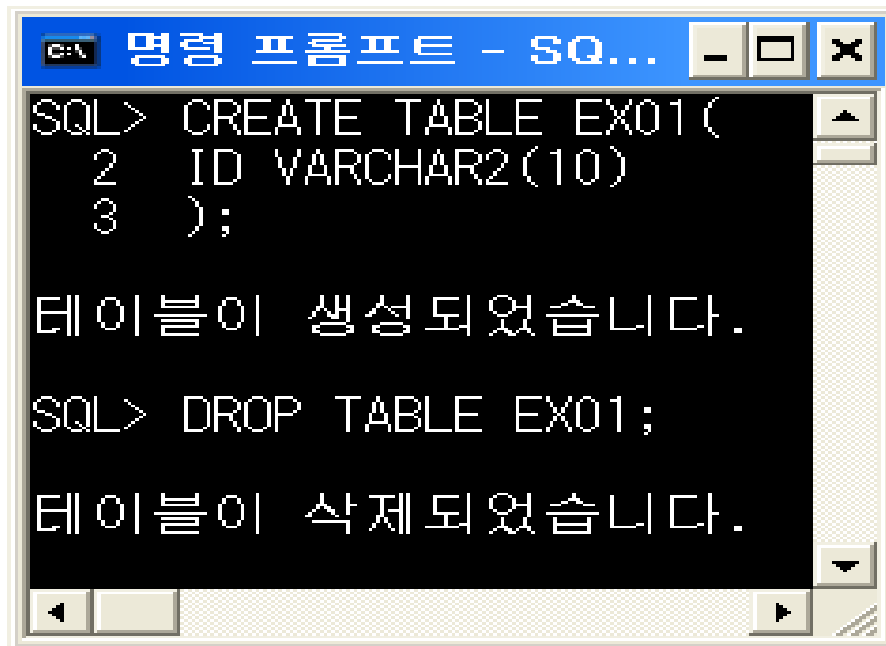


```
C:\ 명령 프롬프트 - SQLPLUS SCOTT/TIGER
SQL> DESCRIBE EMP;
이름                                널?       유형
-----
EMPNO                               NOT NULL   NUMBER(4)
ENAME                               VARCHAR2(10)
JOB                                 VARCHAR2(9)
MGR                                 NUMBER(4)
HIREDATE                           DATE
SAL                                 NUMBER(7,2)
COMM                                NUMBER(7,2)
DEPTNO                             NUMBER(2)
```

DDL - DROP

DROP

DROP TABLE 테이블명



```
c:\ 명령 프롬프트 - SQ...
SQL> CREATE TABLE EX01(
  2   ID VARCHAR2(10)
  3 );

테이블이 생성되었습니다.

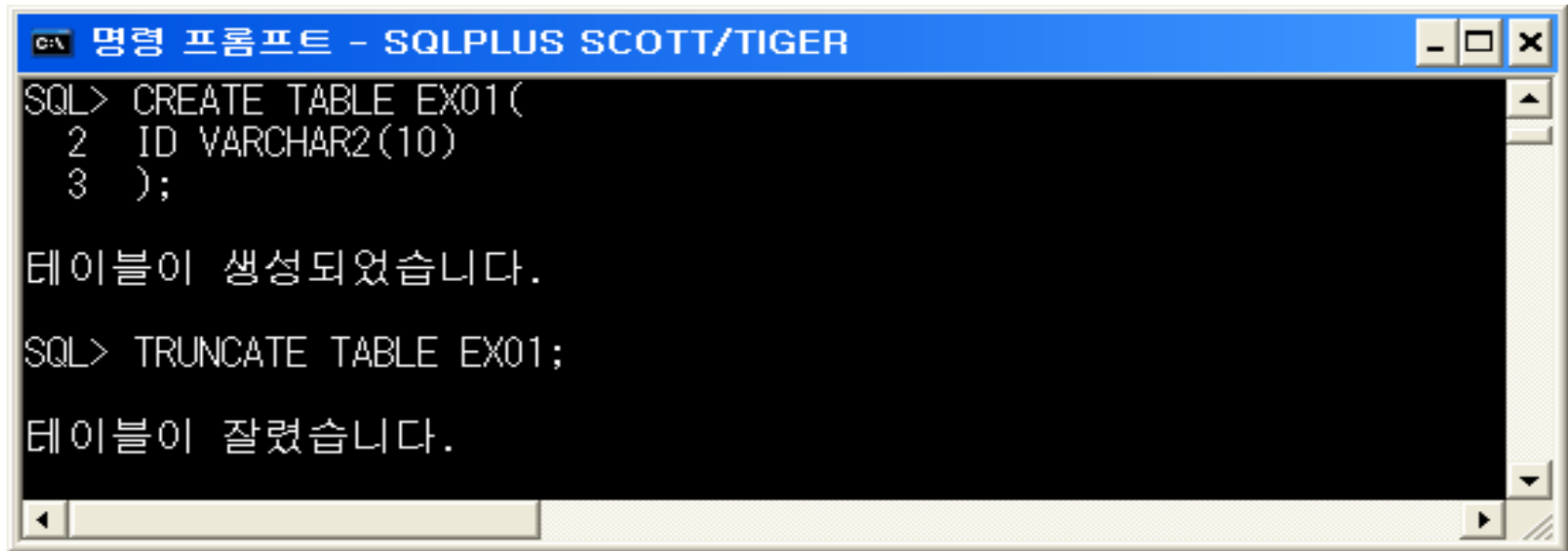
SQL> DROP TABLE EX01;

테이블이 삭제되었습니다.
```

DDL - TRUNCATE

TRUNCATE

TRUNCATE TABLE 테이블명;



```
C:\> 명령 프롬프트 - SQLPLUS SCOTT/TIGER
SQL> CREATE TABLE EX01(
  2  ID VARCHAR2(10)
  3  );

테이블이 생성되었습니다.

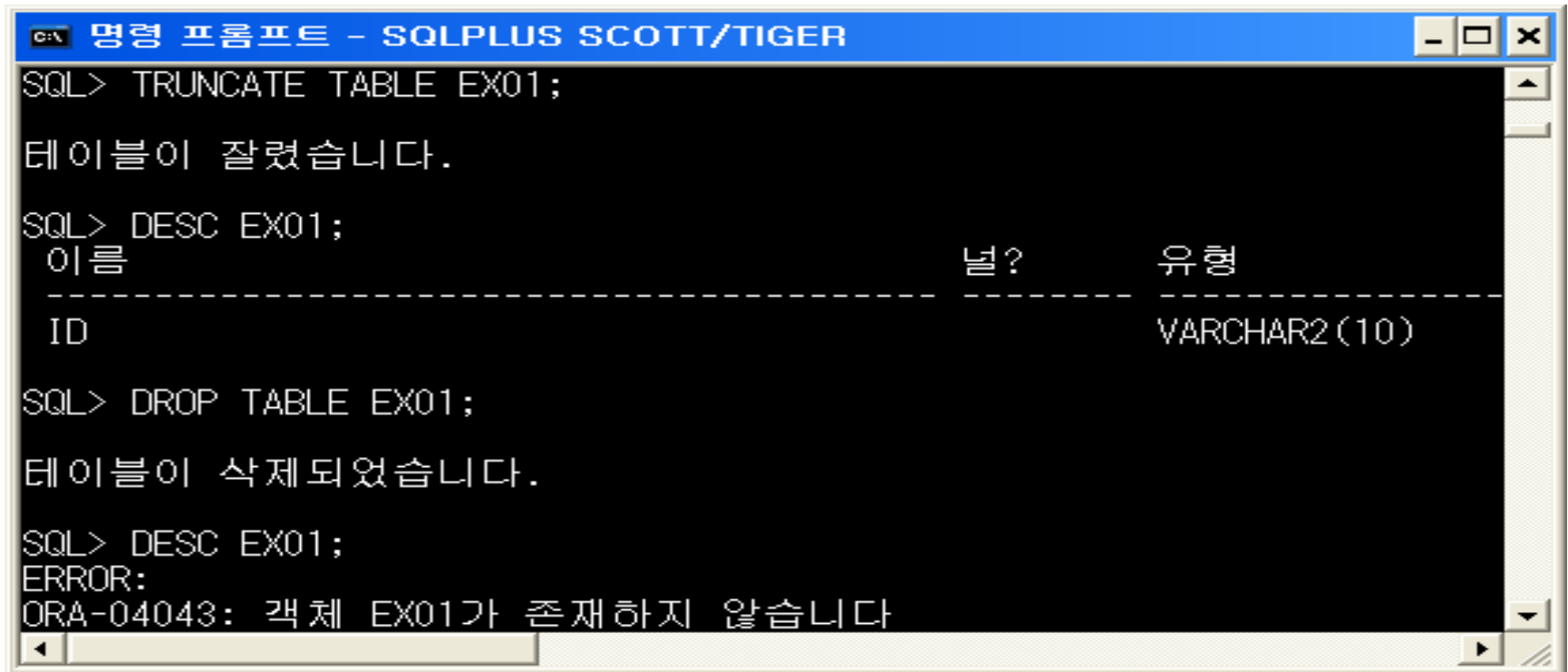
SQL> TRUNCATE TABLE EX01;

테이블이 잘렸습니다.
```

DROP와 TRUNCATE

DROP - 테이블의 구조와 데이터 모두 삭제

TRUNCATE - 테이블의 구조는 남겨두고 데이터만 삭제



```
C:\> 명령 프롬프트 - SQLPLUS SCOTT/TIGER
SQL> TRUNCATE TABLE EX01;
테이블이 잘렸습니다.

SQL> DESC EX01;
  이름                                널?       유형
-----
  ID                                V          VARCHAR2(10)

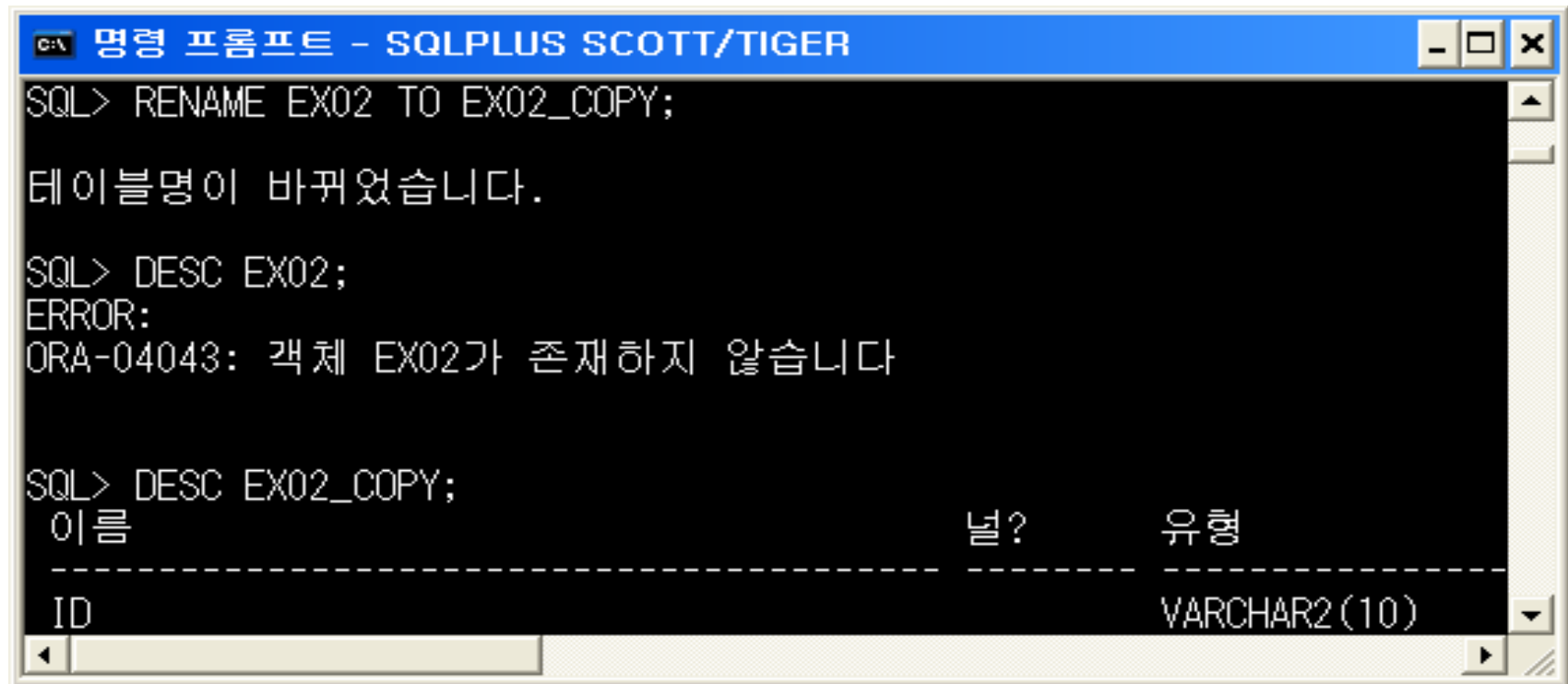
SQL> DROP TABLE EX01;
테이블이 삭제되었습니다.

SQL> DESC EX01;
ERROR:
ORA-04043: 객체 EX01가 존재하지 않습니다
```

DDL - RENAME

RENAME

RENAME 테이블명 **TO** 바꿀 테이블명



```
C:\> 명령 프롬프트 - SQLPLUS SCOTT/TIGER
SQL> RENAME EX02 TO EX02_COPY;

테이블명이 바뀌었습니다.

SQL> DESC EX02;
ERROR:
ORA-04043: 객체 EX02가 존재하지 않습니다

SQL> DESC EX02_COPY;
이름                                널?       유형
-----
ID                                V          VARCHAR2(10)
```

DML - 종류

- **INSERT** (새로운 데이터를 삽입)
- **UPDATE** (기존의 데이터를 변경)
- **DELETE** (기존의 데이터를 삭제)

INSERT INTO 테이블명(컬럼명,)
VALUES(값,);

UPDATE 테이블명
SET 컬럼명 = 변경할 값
[**WHERE** 조건] ;

DELETE
[**FROM**] 테이블명
[**WHERE** 조건] ;

INSERT - 1

INSERT INTO 테이블명(컬럼명,)

VALUES(값1,);

```
C:\ 명령 프롬프트 - sqlplus scott/tiger

SQL> DESC EX01;
이름
-----
ID              NOT NULL  NUMBER(10)
NAME                               VARCHAR2(10)

SQL> INSERT INTO EX01(ID, NAME) VALUES(1, 'S');

1 개의 행이 만들어졌습니다.

SQL> SELECT * FROM EX01;

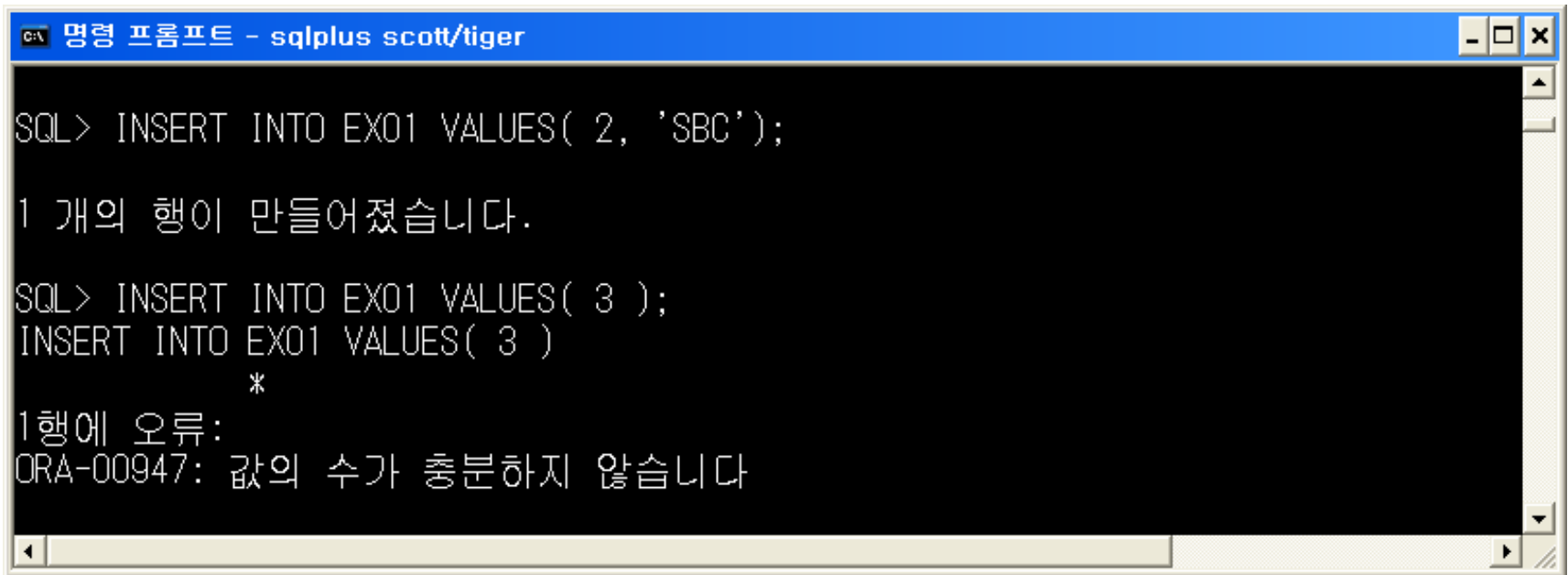
   ID NAME
-----
    1  S

SQL>
```

INSERT - 2

INSERT INTO 테이블명 **VALUES**(값1,);

반드시 테이블의 컬럼의 수와 동일하게 값을 설정해야 한다.



```
C:\ 명령 프롬프트 - sqlplus scott/tiger

SQL> INSERT INTO EX01 VALUES( 2, 'SBC');

1 개의 행이 만들어졌습니다.

SQL> INSERT INTO EX01 VALUES( 3 );
INSERT INTO EX01 VALUES( 3 )
          *
1행에 오류:
ORA-00947: 값의 수가 충분하지 않습니다
```


UPDATE

UPDATE 테이블명 SET 컬럼명 = 바꿀값 WHERE 조건식

C:\ 명령 프롬프트 - sqlplus scott/tiger

```
SQL> SELECT * FROM EX01;
```

ID	NAME
1	S
2	SBC

```
SQL> UPDATE EX01 SET NAME = 'SON';
```

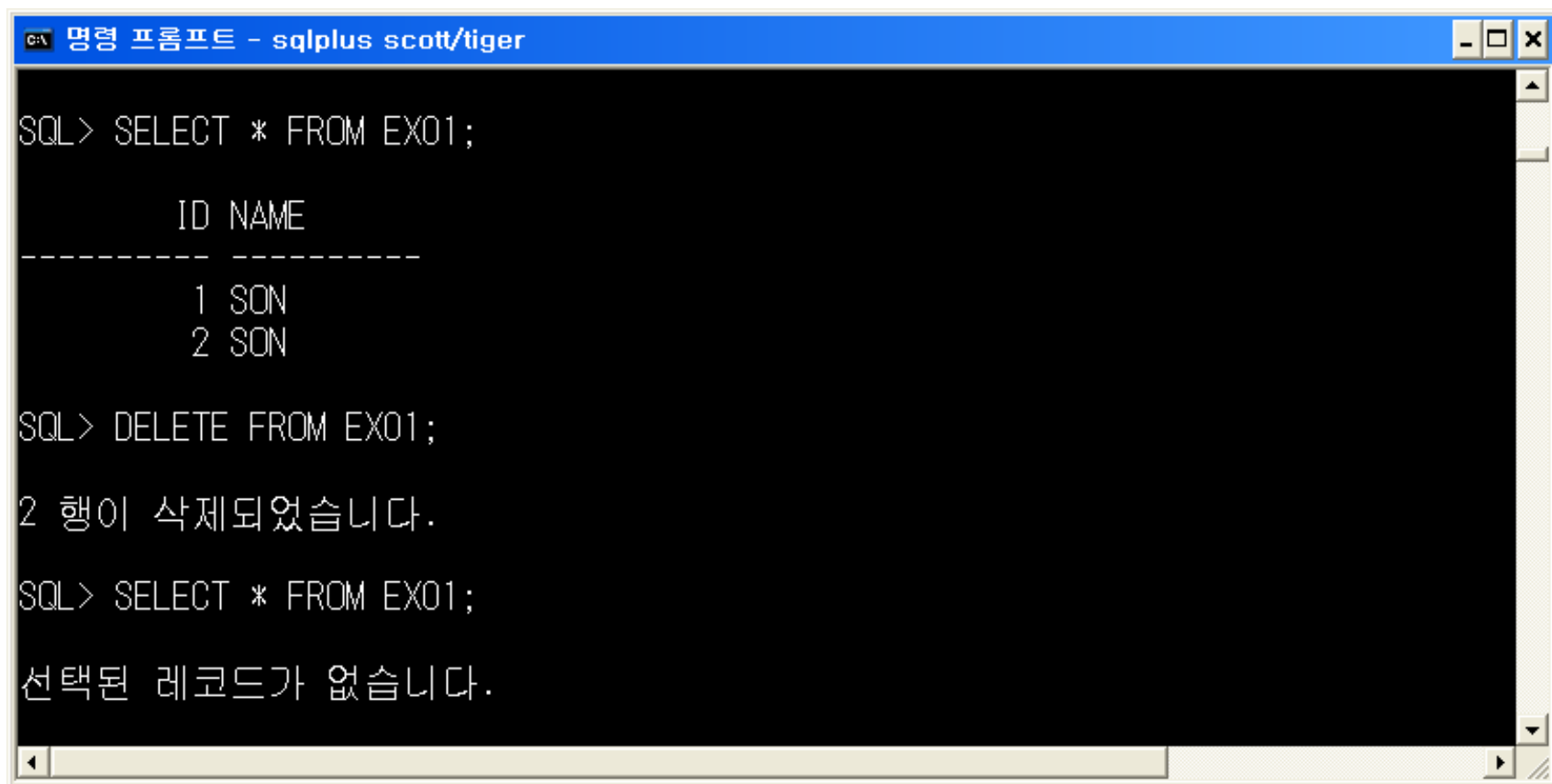
2 행이 갱신되었습니다.

```
SQL> SELECT * FROM EX01;
```

ID	NAME
1	SON
2	SON

DELETE

DELETE [FROM] 테이블명 WHERE 조건식



```
C:\ 명령 프롬프트 - sqlplus scott/tiger

SQL> SELECT * FROM EX01;

      ID NAME
-----
       1 SON
       2 SON

SQL> DELETE FROM EX01;

2 행이 삭제되었습니다.

SQL> SELECT * FROM EX01;

선택된 레코드가 없습니다.
```



SELECT

SELECT 조회할 컬럼명, 조회할 컬럼명

FROM 테이블명;

SELECT EMPNO, ENAME

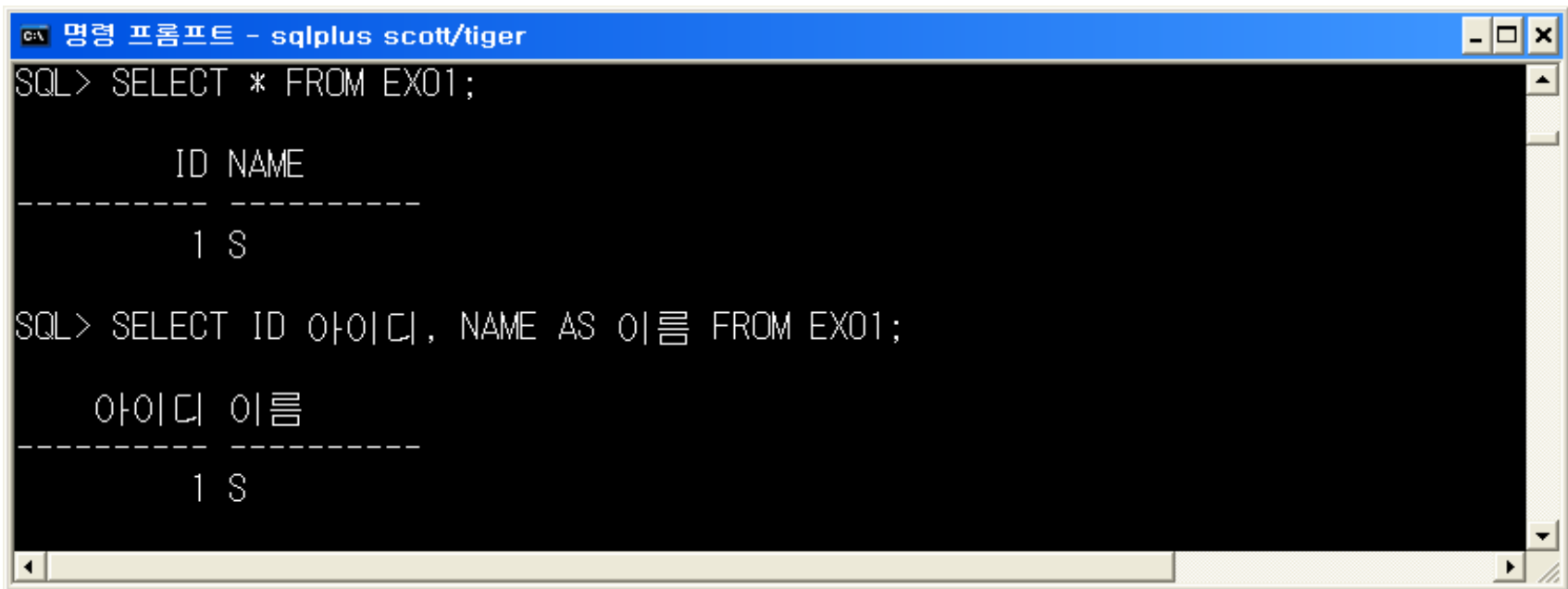
FROM EMP;

1. “*”의 사용 - 테이블에 존재하는 모든 컬럼의 값을 보여준다.
2. ALIAS의 사용 - 화면에 보여지는 컬럼의 헤딩 부분을 바꾼다.
3. 산술 연산자 - 컬럼의 사칙 연산이 가능하다.
4. 합성 연산자 - 컬럼과 문자열의 합성이 가능하다.

ALIAS

ALIAS 의 사용

1. SELECT 컬럼명 AS ALIAS사용 : SELECT ID AS 아이디
2. SELECT 컬럼명 ALIAS사용 : SELECT ID 아이디
3. ALIAS에서 이중부호(" ")의 사용은 공백이나 대소문자 구분 시 사용한다



```
C:\> 명령 프롬프트 - sqlplus scott/tiger
SQL> SELECT * FROM EX01;

   ID NAME
-----
    1 S

SQL> SELECT ID 아이디, NAME AS 이름 FROM EX01;

아이디 이름
-----
    1 S
```



산술연산자

산술연산자의 사용

1. 종류 : +, -, *, /, ()



합성연산자

합성연산자의 사용

1. 2개의 수직 바(||)를 사용한다.
2. 칼럼과 문자 또는 다른 칼럼과 연결 시킨다.
3. 문자 표현식의 결과에 의해 새로운 칼럼의 결과를 보여준다.

WHERE - 조건

SELECT 컬럼명 [ALIAS]

FROM 테이블명

WHERE 조건식;

- **WHERE** 절에 사용되는 연산자

1) 비교 연산자

2) **SQL** 연산자

3) 논리 연산자

4) 조건의 부정

WHERE절에 사용되는 연산자

구 분	연 산 자	연 산 자 의 의 미
논리 연산자	AND	우선 순위 NOT > AND > OR
	OR	
	NOT	
비교 연산자	=	
	>	
	>=	
	<	
	<=	
SQL 비교 연산자	BETWEEN A AND B	A와 B 사이의 값을 찾음
	IN (list)	리스트에 있는 값중 하나라도 일치하면 됨
	LIKE ' 비교문자열 '	비교문자열과 형태가 일치(%, _)
	IS NULL	NULL 값인 지를 판단



비교연산자

비교연산자의 사용

1. 조건을 부여하여 행들을 제한할 때 사용하는 연산자
2. =, >, <, >=, <=
3. CHAR, VARCHAR2와 같은 문자형 타입의 컬럼은 특정 값과 비교하기 위해서는 작은 따옴표(' ')로 묶어서 비교 처리 해야 합니다.



SQL 연산자 - 1

SQL 비교연산자의 사용

1. BETWEEN a AND b : a와 b 사이의 값을 찾습니다.
2. IN(list) : list에 나열되는 값 중 하나만 일치되면 됩니다.
3. LIKE '비교문자열' : 비교 문자열과 형태가 일치하면 됩니다.
4. IS NULL : NULL값을 추출합니다.

SQL 연산자 – BETWEEN ~ AND

BETWEEN 의 사용

1. 컬럼명 BETWEEN a AND b : a가 b 보다 값이 작습니다.

– a(포함)와 b(포함) 사이의 값을 선택

예> salary between 1000 and 2000;

SQL 연산자 - IN

IN 의 사용

1. 컬럼명 IN (값1, 값2, ...)

예>

title in ('부장', '과장');

➔ (title = '부장' or title = '과장') 의미

SQL 연산자 - LIKE

LIKE 의 사용

1. % : 0개 이상의 어떤 문자를 의미합니다.

예> name like '김%';

2. _ : 1개인 단일 문자를 의미합니다.

예> name like '김__';



SQL 연산자 – IS NULL

IS NULL 의 사용

1. NULL 값과의 비교 연산은 거짓(FALSE)을 리턴
2. NULL 값과의 수치 연산은 NULL을 리턴
3. $\text{NULL} + 100 = \text{NULL}$, $\text{NULL} > 180 \Rightarrow \text{FALSE}$



논리 연산자

논리 연산자의 사용

1. 여러 개의 조건들을 논리적으로 연결시키기 위한 연산자
2. AND : 앞, 뒤의 조건이 모두 참일 경우만 참
3. OR : 앞, 뒤의 조건 중 하나만 참이면 참
4. NOT : 뒤에 오는 조건에 반대되는 결과를 돌려 줍니다.

부정연산자

구 분	연 산 자	연 산 자 의 의 미
부정 논리 연산자	!=	같지 않다
	^=	
	<>	
	NOT 컬럼명 =	
	NOT 컬럼명 > A	A 보다 크지 않다
SQL 비교 연산자	NOT BETWEEN A AND B	A와 B 사이에 포함되지 않는 값을 찾는다
	NOT IN (list)	리스트에 일치하지 않는 값을 찾는다
	IS NOT NULL	NULL이 아닌 것을 찾는다

ORDER BY - 정렬

SELECT [DISTINCT] 컬럼명 [ALIAS]

FROM 테이블명

ORDER BY 컬럼명 또는 표현식 [ASC or DESC]

=====

- **ASC** : 조회한 데이터를 오름차순으로 정렬한다(디폴트 값 생략 가능)
- **DESC** : 조회한 데이터를 내림차순으로 정렬한다.
- **ORDER BY** 절은 **SQL**문에서 마지막에 위치한다.
- 컬럼 **ALIAS**명의 사용이 가능하다.
- **NULL** 값은 오름차순일 경우 가장 마지막에 내림차순은 맨 처음 온다.
- 날짜형 데이터는 오름차순일 경우 가장 빠른 날이 먼저 출력된다.
- 기본적인 정렬방식은 오름차순이다.



SQL 기본 함수

1. sysdate : 현재 날짜 정보를 제공
2. to_char : 날짜를 문자로 변경, 숫자를 문자로 변경
3. case 절



SQL 객체 - 시퀀스

1. 오라클에서 제공하는 고유 숫자 생성 객체
2. `create sequence` 객체명;
3. 객체명.currVal : 마지막 생성된 고유번호
4. 객체명.nextVal : 고유번호 추출

1. 지정한 수만큼의 상위 데이터만 조회
 - rownum 사용