

틀리기 쉬운 SQL

컴퓨터공학과 학생의 학번과 이름을 찾아라.

➤ **wrong answer**

SQL Plus

```
SQL> select stu_id, name  
2   from student  
3   where dept_id = '920';
```

STU_ID	NAME
1292001	김광식
1292002	김정현
1292003	김현정

```
SQL> _
```

컴퓨터공학과 학생의 학번과 이름을 찾아라.

➤ correct answer

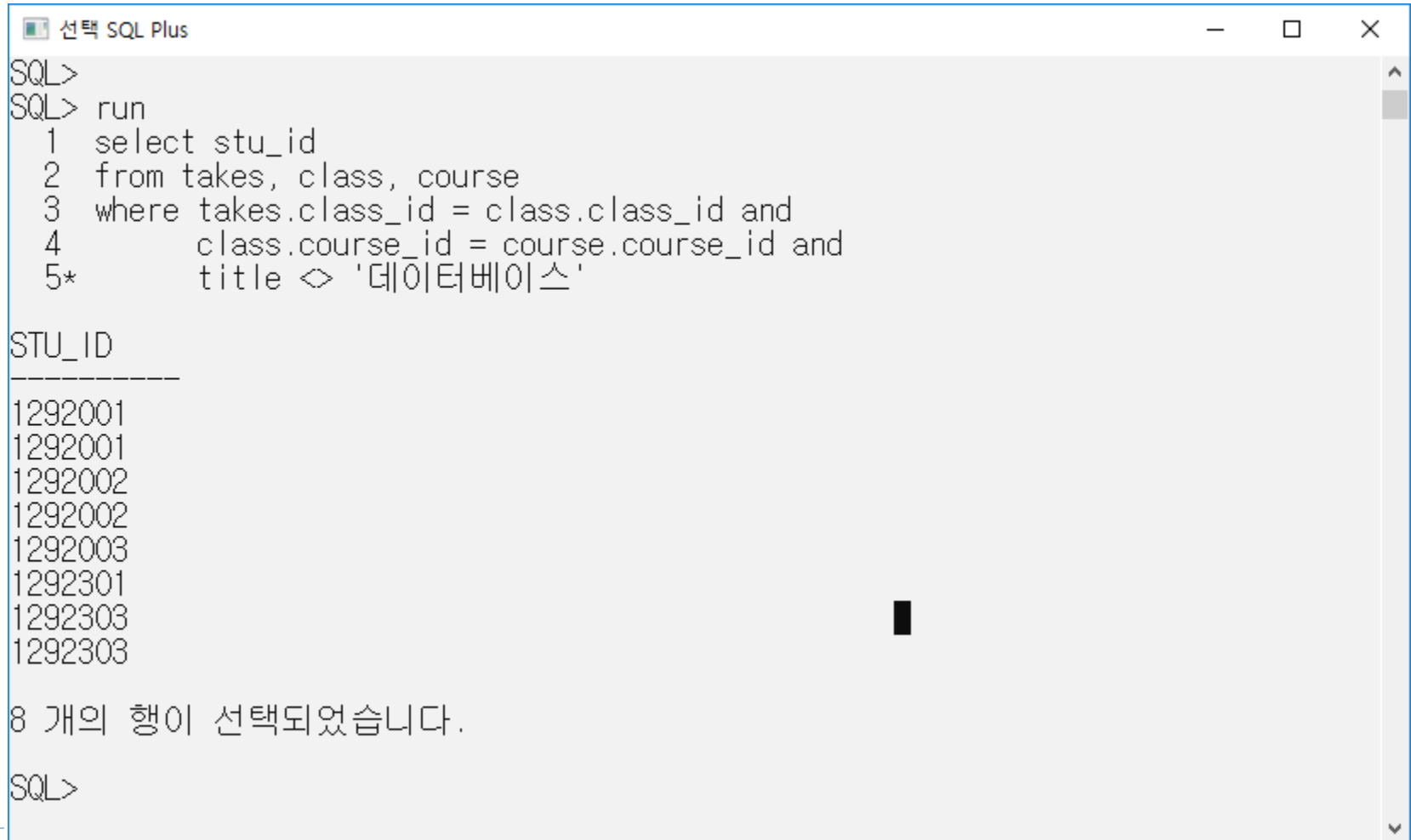
SQL Plus

```
SQL>
SQL> select stu_id, name
  2   from student s, department d
  3  where s.dept_id = d.dept_id and
  4  dept_name = '컴퓨터공학과';
```

STU_ID	NAME
1292001	김광식
1292002	김정현
1292003	김현정

데이터베이스를 수강하지 않는 학생의 학번을 찾아라

➤ wrong answer



```
선택 SQL Plus
SQL>
SQL> run
  1 select stu_id
  2 from takes, class, course
  3 where takes.class_id = class.class_id and
  4        class.course_id = course.course_id and
  5*    title <> '데이터베이스'

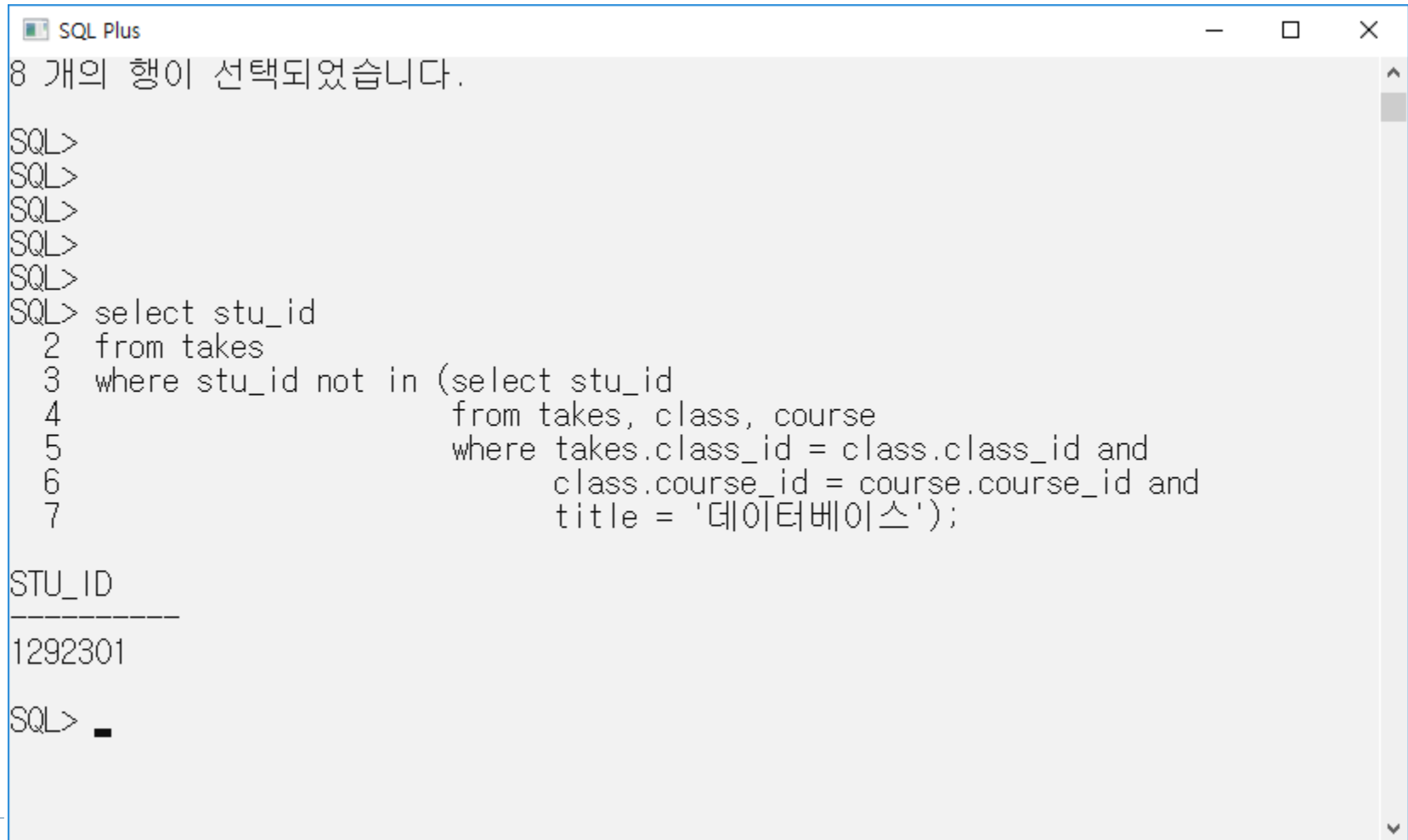
STU_ID
-----
1292001
1292001
1292002
1292002
1292003
1292301
1292303
1292303

8 개의 행이 선택되었습니다.

SQL>
```

데이터베이스를 수강하지 않는 학생의 학번을 찾아라

➤ wrong answer



```
SQL Plus
8 개의 행이 선택되었습니다.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> select stu_id
  2  from takes
  3  where stu_id not in (select stu_id
  4                        from takes, class, course
  5                        where takes.class_id = class.class_id and
  6                            class.course_id = course.course_id and
  7                            title = '데이터베이스');

STU_ID
-----
1292301

SQL> _
```

데이터베이스를 수강하지 않는 학생의 학번을 찾아라

➤ correct answer

```
선택 SQL Plus
SQL>
SQL> run
  1  select stu_id
  2  from student
  3  where stu_id not in ( select stu_id
  4                        from takes, class, course
  5                        where takes.class_id = class.class_id and
  6                              class.course_id = course.course_id and
  7*                               title = '데이터베이스')

STU_ID
-----
1292501
1292301
1292502
1292305

SQL>
```

데이터베이스, 운영체제 중 적어도 한 과목을 수강한 학생의 이름을 찾아라.

➤ **wrong answer**

```
SQL> select distinct s.name
      2  from student s, takes t, class cl, course co
      3  where s.stu_id = t.stu_id and
      4  t.class_id = cl.class_id and
      5  cl.course_id = co.course_id and
      6  title = '데이터베이스' or title = '운영체제';
```

NAME

김광식
김정현
김현정
박광수
김우주
박철수
백태성

7 행이 선택되었습니다.

AND 연산자는 OR 연산자보다 우선순위가 높음

6. 데이터베이스, 운영체제 중 적어도 한 과목을 수강한 학생의 이름을 찾아라.

➤ correct answer

```
SQL Plus

SQL> select distinct s.name
  2  from student s, takes t, class cl, course co
  3  where s.stu_id = t.stu_id and
  4    t.class_id = cl.class_id and
  5    cl.course_id = co.course_id and
  6    ((title = '데이터베이스' or title = '운영체제');
```

NAME

김광식
김정현
김현정
박광수

데이터베이스, 운영체제 중 적어도 한 과목을 수강한 학생의 이름을 찾아라.

➤ **union**

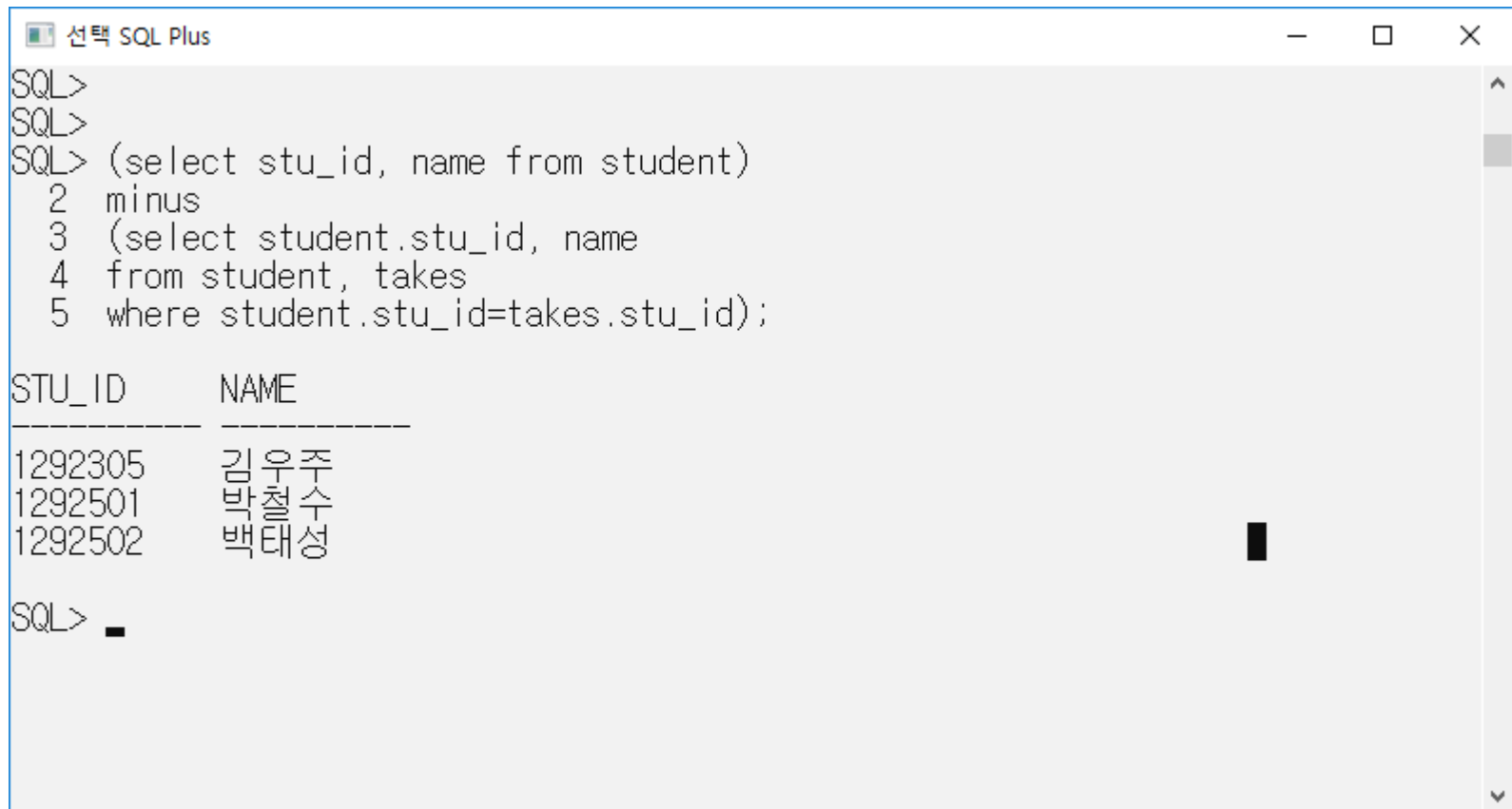
```
선택 SQL Plus
SQL>
SQL> (select s.name
  2   from student s, takes t, class cl, course co
  3   where s.stu_id = t.stu_id and
  4         t.class_id = cl.class_id and
  5         cl.course_id = co.course_id and
  6         title = '데이터베이스')
  7 union
  8 (select s.name
  9   from student s, takes t, class cl, course co
 10  where s.stu_id = t.stu_id and
 11        t.class_id = cl.class_id and
 12        cl.course_id = co.course_id and
 13        title = '운영체제');

NAME
-----
김광식
김정현
김현정
김광수
박광수

SQL>
```

9. 한 과목도 수강하지 않은 학생의 학번과 이름을 찾아라.

➤ 차집합



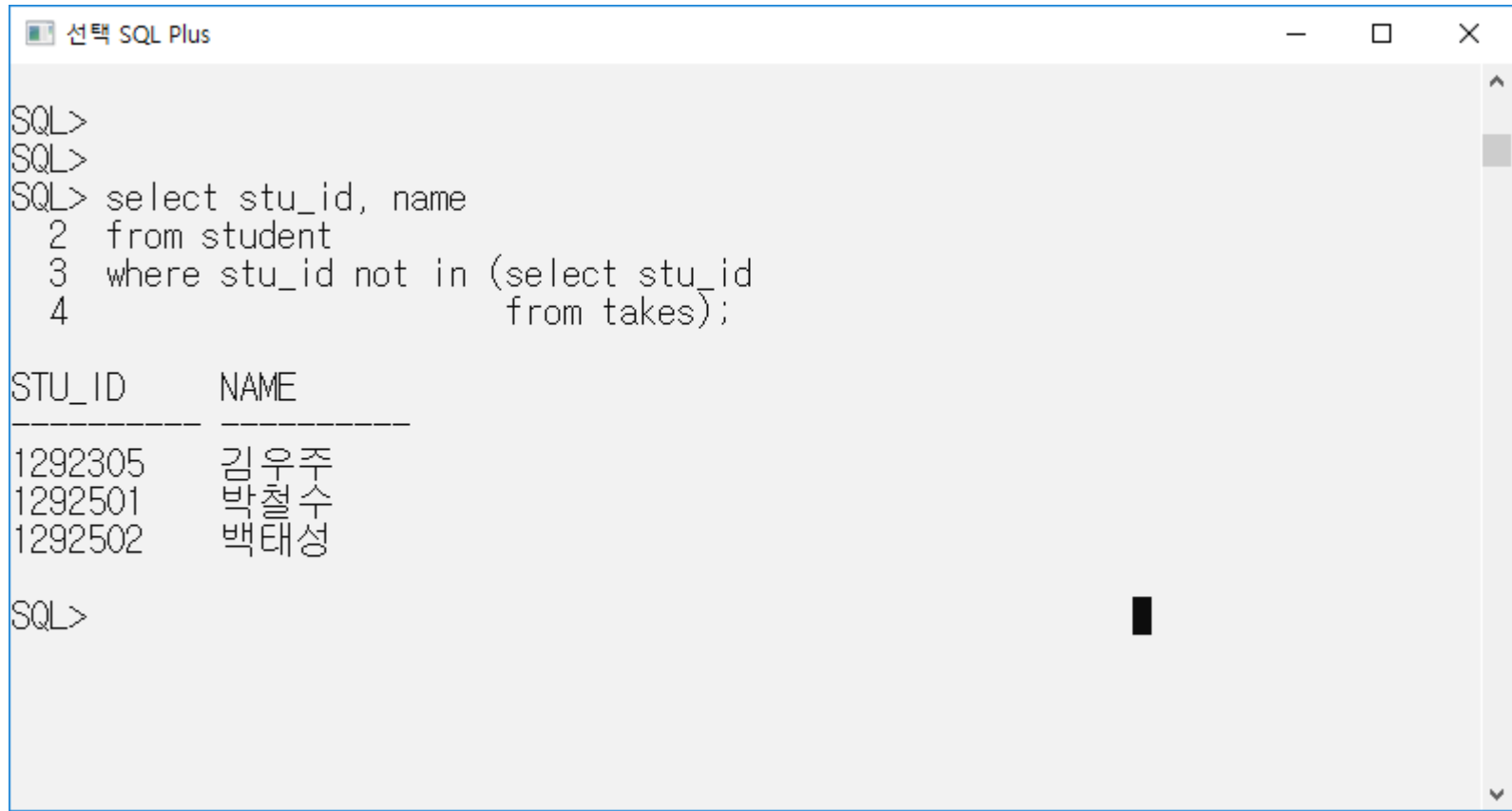
```
선택 SQL Plus
SQL>
SQL>
SQL> (select stu_id, name from student)
  2  minus
  3  (select student.stu_id, name
  4    from student, takes
  5   where student.stu_id=takes.stu_id);

STU_ID      NAME
-----
1292305     김우주
1292501     박철수
1292502     백태성

SQL> _
```

한 과목도 수강하지 않은 학생의 학번과 이름을 찾아라.

➤ **not in**



The screenshot shows a SQL Plus window titled "선택 SQL Plus". The prompt "SQL>" is visible. The query entered is:

```
SQL> select stu_id, name
2   from student
3  where stu_id not in (select stu_id
4                        from takes);
```

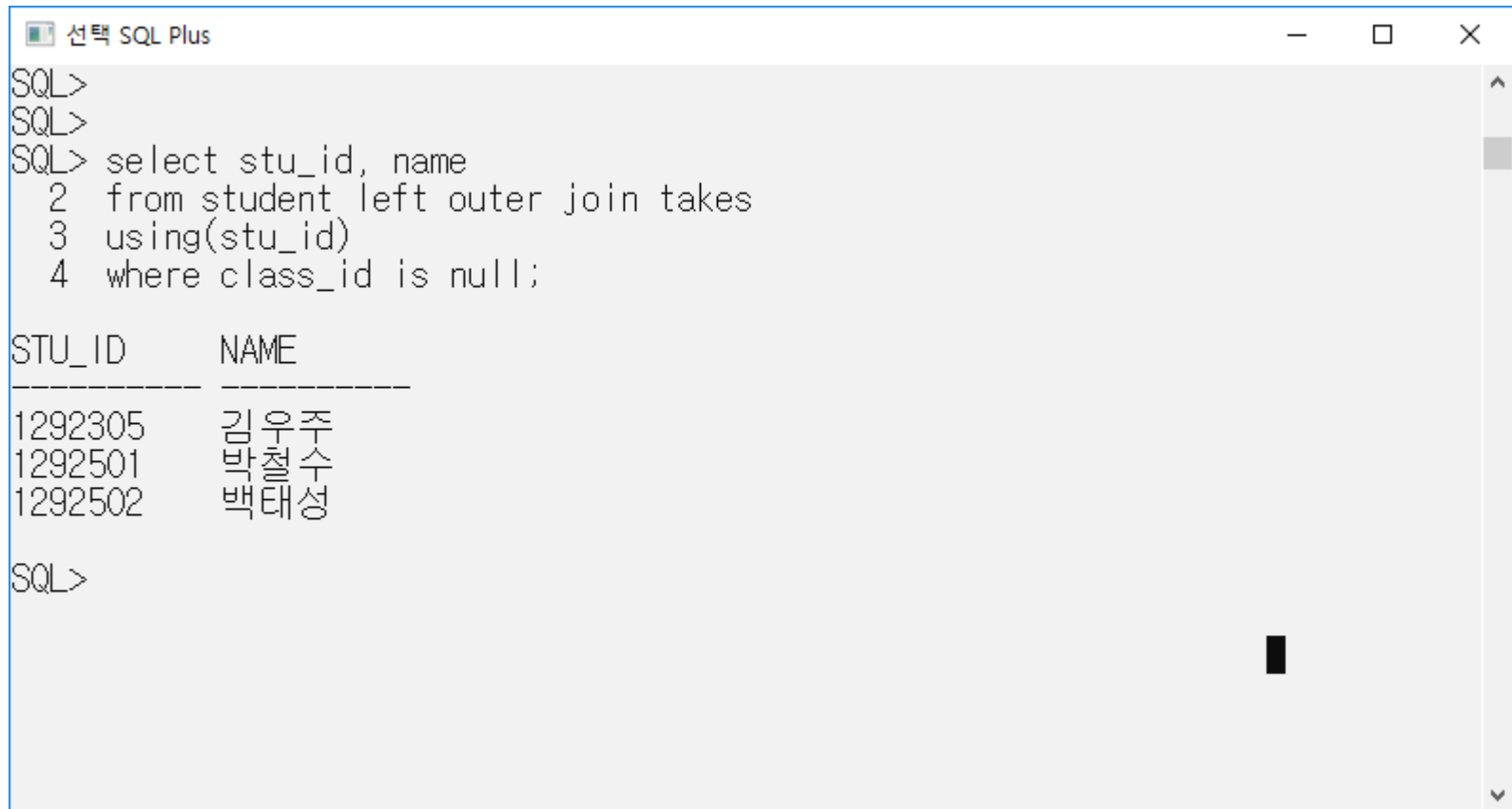
The result is displayed as a table with two columns: STU_ID and NAME. The data rows are:

STU_ID	NAME
1292305	김우주
1292501	박철수
1292502	백태성

The prompt "SQL>" is visible again at the bottom of the window.

한 과목도 수강하지 않은 학생의 학번과 이름을 찾아라.

➤ **left outer join**



The screenshot shows a SQL Plus window titled "선택 SQL Plus". The command prompt shows the following SQL query:

```
SQL>
SQL>
SQL> select stu_id, name
2  from student left outer join takes
3  using(stu_id)
4  where class_id is null;
```

The query results are displayed in a table format:

STU_ID	NAME
1292305	김우주
1292501	박철수
1292502	백태성

The window also shows the prompt "SQL>" at the bottom, indicating the query has been executed.

takes 테이블에 score 필드를 char(1) 타입으로 추가하라.

선택 SQL Plus

```
SQL>
SQL> alter table takes
      2  add score char(1);
```

테이블이 변경되었습니다.

```
SQL> select * from takes;
```

STU_ID	CLASS_ID	GRADE S
1292001	C101-01	85
1292001	C103-01	98
1292001	C301-01	91
1292002	C301-01	92
1292002	C103-01	86
1292002	C502-02	77
1292003	C103-02	80
1292003	C501-02	95
1292301	C102-01	78
1292303	C102-01	71
1292303	C103-02	88

STU_ID	CLASS_ID	GRADE S
1292303	C501-01	98

12 개의 행이 선택되었습니다.

```
SQL> █
```

takes

```
선택 SQL Plus
SQL>
SQL>
SQL> desc takes;
이름                널?       유형
-----
STU_ID              NOT NULL  VARCHAR2(10)
CLASS_ID            NOT NULL  VARCHAR2(10)
GRADE               NUMBER(38)
SCORE               CHAR(1)

SQL> █
```

Score 필드에 성적의 등급을 부여하라. (A, B, C, F)

➤ case

```
선택 SQL Plus
SQL>
SQL>
SQL> update takes
  2   set score =
  3           (case when grade >= 90 then 'A'
  4                  when grade between 80 and 89 then 'B'
  5                  when grade between 70 and 79 then 'C'
  6                  else 'F'
  7              end);

12 행이 갱신되었습니다.

SQL> select * from takes;
```

STU_ID	CLASS_ID	GRADE	S
1292001	C101-01	85	B
1292001	C103-01	98	A
1292001	C301-01	91	A
1292002	C301-01	92	A
1292002	C103-01	86	B
1292002	C502-02	77	C
1292003	C103-02	80	B
1292003	C501-02	95	A
1292301	C102-01	78	C
1292303	C102-01	71	C
1292303	C103-02	88	B

STU_ID	CLASS_ID	GRADE	S
1292303	C501-01	98	A

```

12 개의 행이 선택되었습니다.
```

takes 테이블에서 score 필드를 삭제하라.

선택 SQL Plus

```
SQL> alter table takes  
2 drop column score;
```

테이블이 변경되었습니다.

```
SQL> select * from takes;
```

STU_ID	CLASS_ID	GRADE
1292001	C101-01	85
1292001	C103-01	98
1292001	C301-01	91
1292002	C301-01	92
1292002	C103-01	86
1292002	C502-02	77
1292003	C103-02	80
1292003	C501-02	95
1292301	C102-01	78
1292303	C102-01	71
1292303	C103-02	88

STU_ID	CLASS_ID	GRADE
1292303	C501-01	98

12 개의 행이 선택되었습니다.

```
SQL>
```


15. 한글 길이

length()는 글자수(한글 한자도 한글자), lengthb는 바이트 수
||g standard 및 2|c는 한글 한글자 3바이트, ||g express는 2바이트

```
Run SQL Command Line

SQL>
SQL> select title, length(title), lengthb(title)
  2  from course;

TITLE                                LENGTH(TITLE)  LENGTHB(TITLE)
-----
전산개론                             4              12
자료구조                             4              12
데이터베이스                         6              18
운영체제                             4              12
컴퓨터구조                           5              15
이산수학                             4              12
객체지향언어                         6              18
인공지능                             4              12
알고리즘                             4              12

9 rows selected.

SQL> _
```

15. DB 캐릭터셋 확인방법

SQL Plus

SQL*Plus: Release 21.0.0.0.0 - Production on 화 3월 29 12:38:05 2022
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

사용자명 입력: yukim

비밀번호 입력:

마지막 성공한 로그인 시간: 화 3월 29 2022 12:35:44 +09:00

다음에 접속됨:

Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> select * from nls_database_parameters where parameter = 'NLS_CHARACTERSET';

PARAMETER

VALUE

NLS_CHARACTERSET

AL32UTF8

KO16KSC5601 : 완성형 한글(한글 바이트 : 2byte)

KO16MSWIN949 : 조합형 한글(한글 바이트 : 2byte)

AL32UTF8 : 유니코드의 CES중의 한 부분(한글 바이트: 3byte)

DATE

SQL> Run SQL Command Line

```
SQL> create table emp
2  (
3      empno          number(4),
4      ename          varchar2(10)          not null,
5      hiredate       date,
6      sal            number(10),
7      constraint pk_emp primary key(empno)
8  );
```

Table created.

```
SQL> _
```

DATE

➤ 날짜 입력

Run SQL Command Line

```
SQL> insert into emp  
2 values(9999, '황진이', '21/04/21', 2000000);
```

```
1 row created.
```

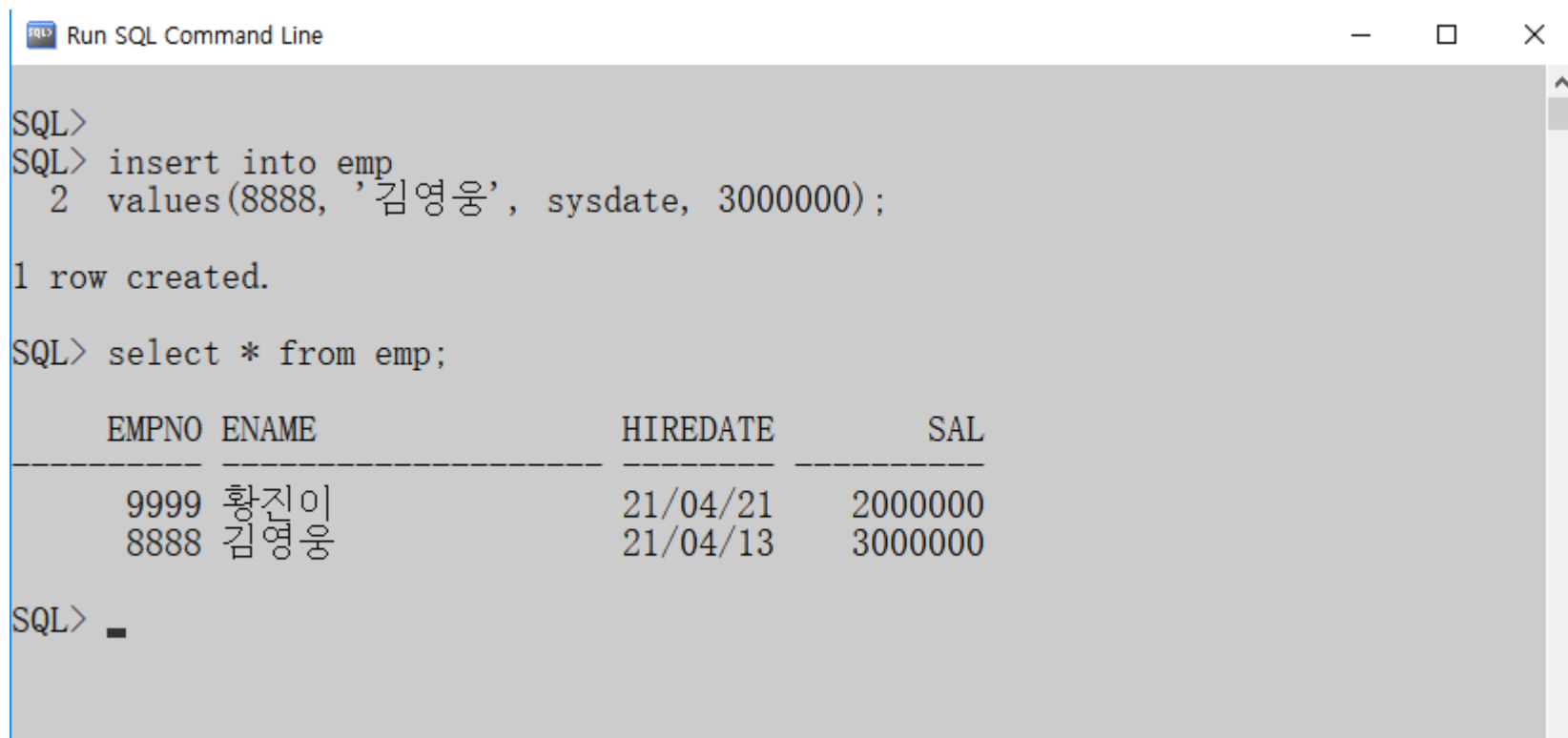
```
SQL> select * from emp;
```

EMPNO	ENAME	HIREDATE	SAL
9999	황진이	21/04/21	2000000

```
SQL> _
```

DATE

➤ sysdate



The screenshot shows a SQL Command Line window titled "Run SQL Command Line". The window contains the following text:

```
SQL>
SQL> insert into emp
  2 values(8888, '김영웅', sysdate, 3000000);

1 row created.

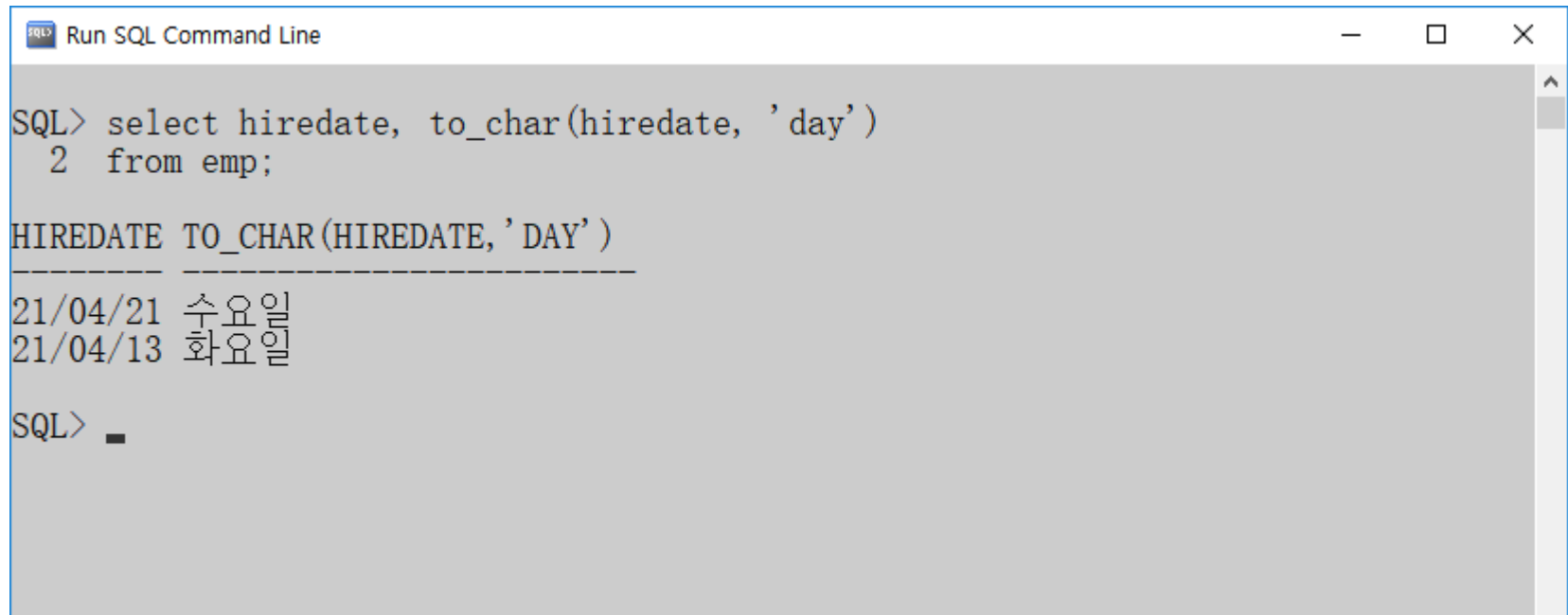
SQL> select * from emp;
```

EMPNO	ENAME	HIREDATE	SAL
9999	황진이	21/04/21	2000000
8888	김영웅	21/04/13	3000000

SQL> █

DATE

➤ 요일



```
SQL> select hiredate, to_char(hiredate, 'day')
2  from emp;

HIREDATE TO_CHAR(HIREDATE, 'DAY')
-----
21/04/21 수요일
21/04/13 화요일

SQL> _
```