

데이터베이스

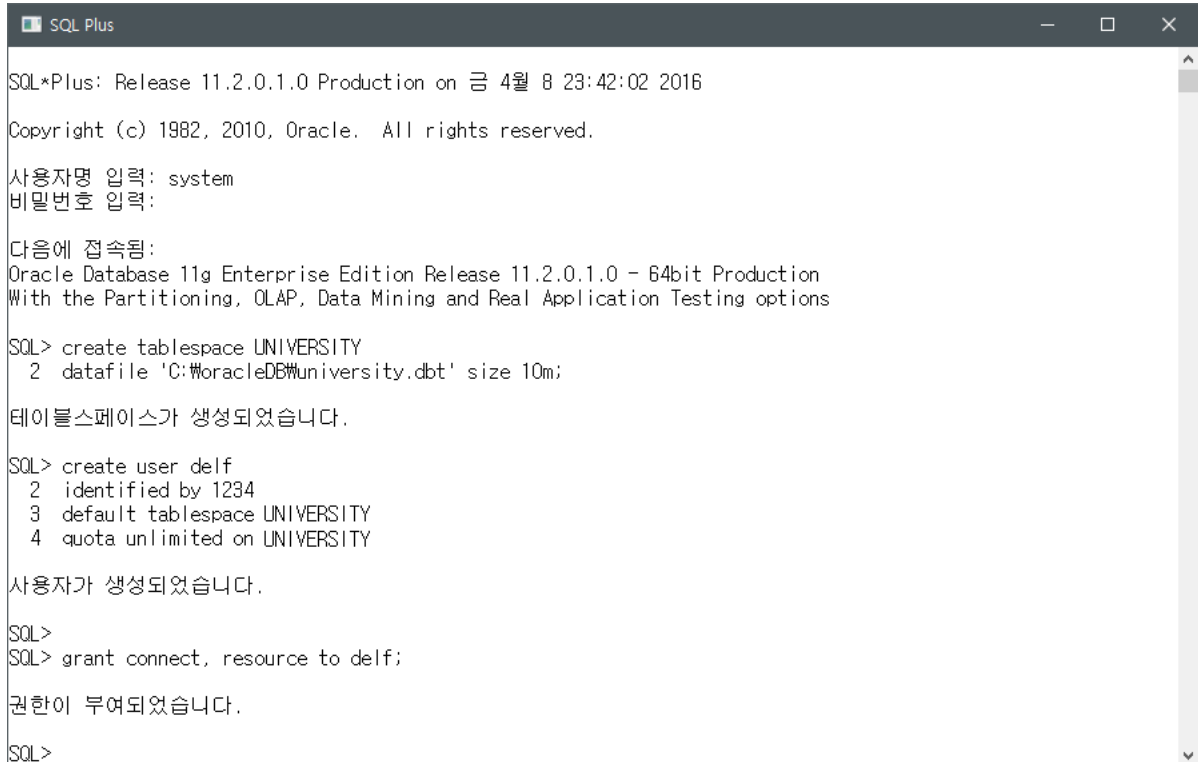
Report #2

- Oracle SQL -

이 름 : 이 상 훈
학 번 : 1 2 9 2 0 3 2
분 반 : B
담당 교수 : 김영웅 교수님
제 출 일 : 2016. 04. 14

<문제 1>

- TABLESPACE를 생성한 후, 이를 사용할 USER를 생성하고 권한을 부여한다.



```

SQL*Plus: Release 11.2.0.1.0 Production on 금 4월 8 23:42:02 2016

Copyright (c) 1982, 2010, Oracle. All rights reserved.

사용자명 입력: system
비밀번호 입력:

다음에 접속됨:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> create tablespace UNIVERSITY
  2  datafile 'C:\oracledb\university.dbf' size 10m;

테이블스페이스가 생성되었습니다.

SQL> create user delf
  2  identified by 1234
  3  default tablespace UNIVERSITY
  4  quota unlimited on UNIVERSITY

사용자가 생성되었습니다.

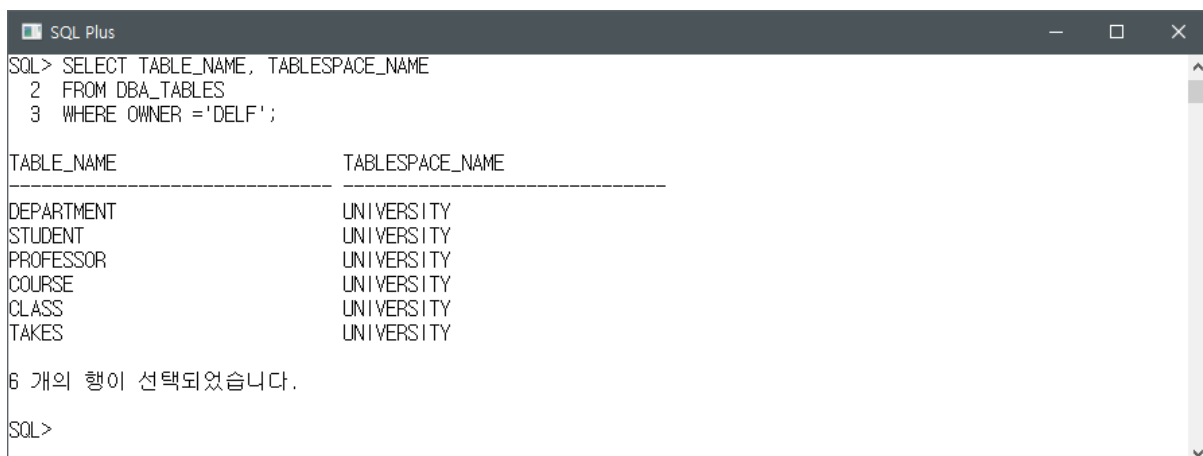
SQL>
SQL> grant connect, resource to delf;

권한이 부여되었습니다.

SQL>
  
```

<그림1-A> TABLESPACE 생성 및 USER생성과 권한 부여

- 권한을 가진 USER DELFL로 TABLESPACE UNIVERSITY에 테이블을 생성한다.
생성한 테이블들은 아래와 같다.



```

SQL> SELECT TABLE_NAME, TABLESPACE_NAME
  2  FROM DBA_TABLES
  3  WHERE OWNER = 'DELFL';

TABLE_NAME                                TABLESPACE_NAME
-----
DEPARTMENT                                UNIVERSITY
STUDENT                                  UNIVERSITY
PROFESSOR                                UNIVERSITY
COURSE                                  UNIVERSITY
CLASS                                  UNIVERSITY
TAKES                                  UNIVERSITY

6 개의 행이 선택되었습니다.

SQL>
  
```

<그림1-B> DELFL가 생성한 TABLE 목록

문제1-1. 컴퓨터 공학과 학생의 학번과 이름을 찾아라.

select name, stu_id

from student;



```
SQL> select name, stu_id
2  from student;
```

NAME	STU_ID
김광식	1292001
김정현	1292002
김현정	1292003
김현정	1292301
박광수	1292303
김우주	1292305
박철수	1292501
백태성	1292502
이상훈	1292032

9 개의 행이 선택되었습니다.

```
SQL>
```

<그림 1-1> 문제 1-1 의 SQL 문과 그 결과

문제1-2. 과목명에 '구조'가 들어 있는 과목번호와 과목명을 찾아라.

select course_id, title

from course

where title **like** '%구조%';



```
SQL> select course_id, title
2  from course
3  where title like '%구조%';
```

COURSE_ID	TITLE
C102	자료구조
C302	컴퓨터구조

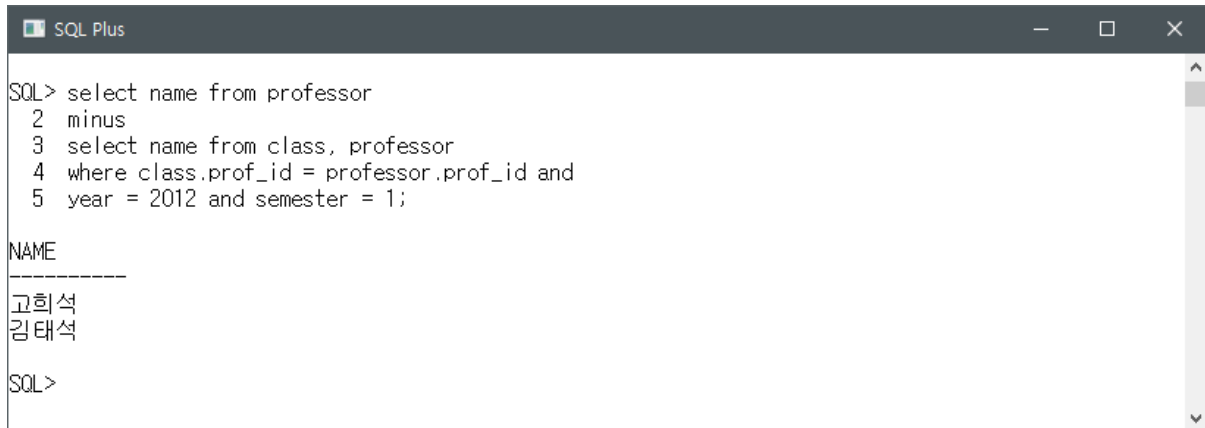
```
SQL> █
```

<그림 1-2> 문제 1-2 의 SQL 문과 그 결과

문제1-3. 2012 년도 1 학기에 강의가 없는 교수의 이름을 찾아라.

(1) 집합연산 minus 사용

```
select name from professor
minus
select name from class, professor
where class.prof_id = professor.prof_id
and year = 2012 and semester = 1;
```



The screenshot shows a SQL Plus window with the following content:

```
SQL> select name from professor
2 minus
3 select name from class, professor
4 where class.prof_id = professor.prof_id and
5 year = 2012 and semester = 1;

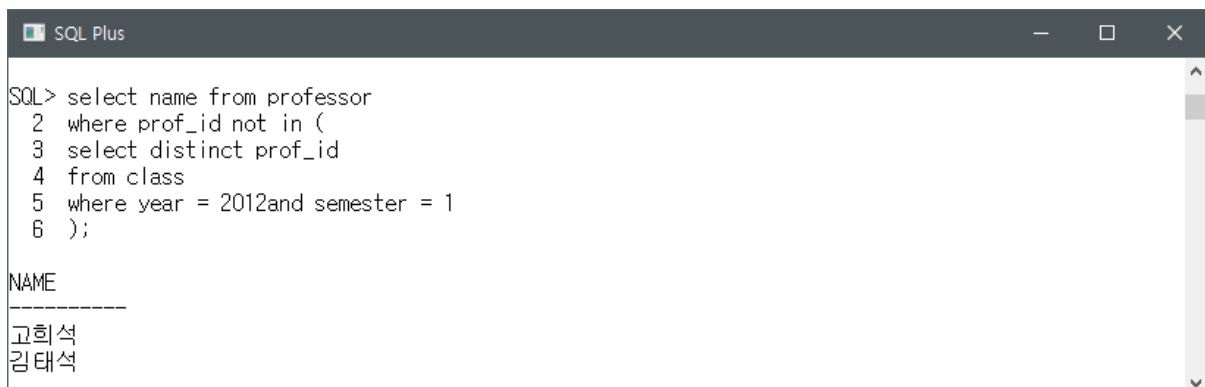
NAME
-----
교회석
김태석

SQL>
```

<그림 1-3-(1)> 문제 1-3(1)의 SQL 문과 그 결과

(2) 중첩질의 not in 사용

```
select name from professor
where prof_id not in (
    select distinct prof_id
    from class
    where year = 2012and semester = 1
);
```



The screenshot shows a SQL Plus window with the following content:

```
SQL> select name from professor
2 where prof_id not in (
3 select distinct prof_id
4 from class
5 where year = 2012and semester = 1
6 );

NAME
-----
교회석
김태석
```

<그림 1-3-(2)> 문제 1-3(2)의 SQL 문과 그 결과

문제1-4. 2012 년도 1 학기에 한 과목도 수강하지 않은 학생의 학번과 이름, 학과명을 찾아라

(1) 집합연산 minus 사용

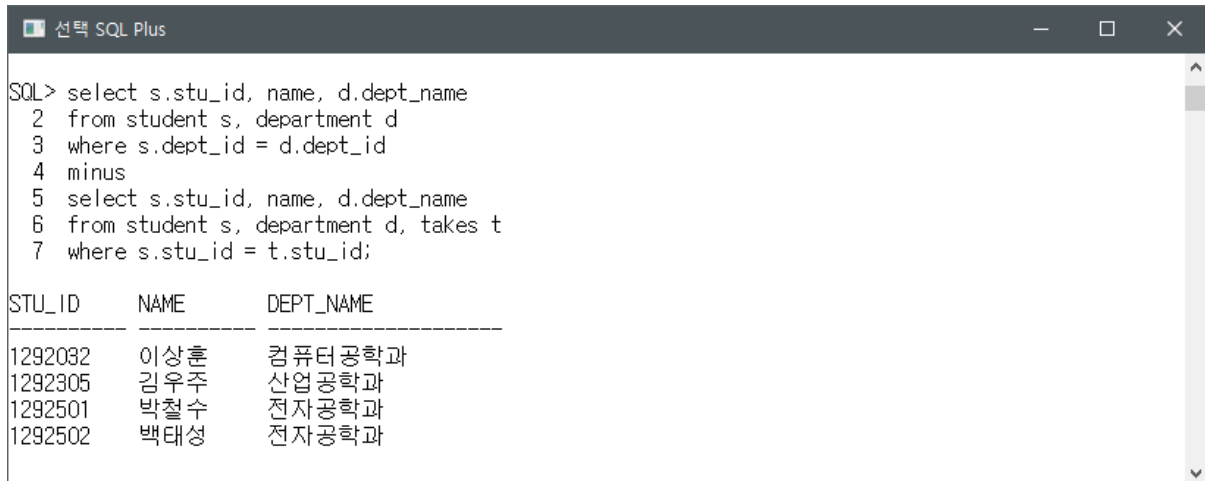
select s.stu_id, name, d.dept_name **from** student s, department d

where s.dept_id = d.dept_id

minus

select s.stu_id, name, d.dept_name **from** student s, department d, takes t

where s.stu_id = t.stu_id;



선택 SQL Plus

```
SQL> select s.stu_id, name, d.dept_name
2  from student s, department d
3  where s.dept_id = d.dept_id
4  minus
5  select s.stu_id, name, d.dept_name
6  from student s, department d, takes t
7  where s.stu_id = t.stu_id;
```

STU_ID	NAME	DEPT_NAME
1292032	이상훈	컴퓨터공학과
1292305	김우주	산업공학과
1292501	박철수	전자공학과
1292502	백태성	전자공학과

<그림 1-4-(1)> 문제 1-4(1)의 SQL 문과 그 결과

(2) 중첩질의 not in 사용

select stu_id, name, dept_name **from** student, department

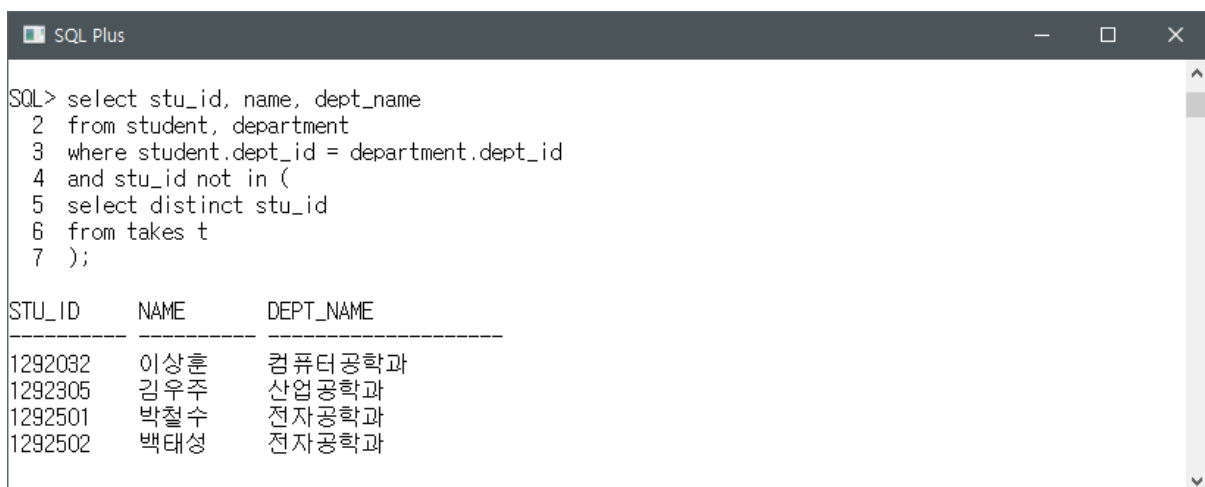
where student.dept_id = department.dept_id

and stu_id **not in** (

select distinct stu_id

from takes t

);



SQL Plus

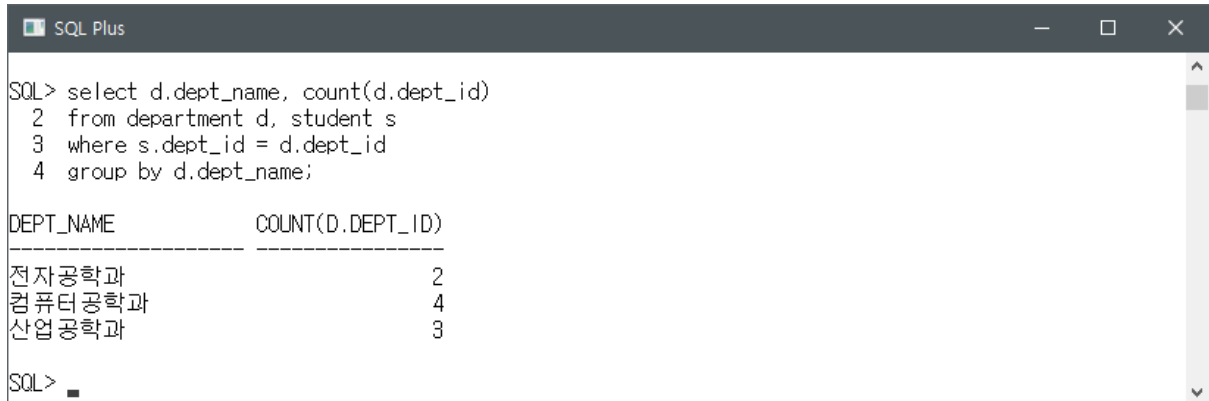
```
SQL> select stu_id, name, dept_name
2  from student, department
3  where student.dept_id = department.dept_id
4  and stu_id not in (
5  select distinct stu_id
6  from takes t
7  );
```

STU_ID	NAME	DEPT_NAME
1292032	이상훈	컴퓨터공학과
1292305	김우주	산업공학과
1292501	박철수	전자공학과
1292502	백태성	전자공학과

<그림 1-4-(2)> 문제 1-4(2)의 SQL 문과 그 결과

문제1-5. 학과별 학생 수를 찾아라. (학과명, 학생수)

```
select d.dept_name, count(d.dept_id)
from department d, student s
where s.dept_id = d.dept_id
group by d.dept_name;
```



SQL Plus window showing the execution of the SQL query. The query is: select d.dept_name, count(d.dept_id) from department d, student s where s.dept_id = d.dept_id group by d.dept_name; The results are displayed in a table with two columns: DEPT_NAME and COUNT(D.DEPT_ID). The results are: 전자공학과 (2), 컴퓨터공학과 (4), and 산업공학과 (3).

DEPT_NAME	COUNT(D.DEPT_ID)
전자공학과	2
컴퓨터공학과	4
산업공학과	3

<그림 1-5> 문제 1-5 의 SQL 문과 그 결과

문제1-6. 학번별 수강과목 수를 찾아라. (학번, 수강 과목 수)

```
select stu_id, count(stu_id)
from takes
group by stu_id;
```



SQL Plus window showing the execution of the SQL query. The query is: select stu_id, count(stu_id) from takes group by stu_id; The results are displayed in a table with two columns: STU_ID and COUNT(STU_ID). The results are: 1292001 (3), 1292002 (3), 1292003 (2), 1292301 (1), and 1292303 (3).

STU_ID	COUNT(STU_ID)
1292001	3
1292002	3
1292003	2
1292301	1
1292303	3

<그림 1-6> 문제 1-6 의 SQL 문과 그 결과

문제1-7. 가장 최근에 임용된 교수의 이름과 재직 연수를 찾아라.

(1) max 또는 min 사용

```
select name 이름, 2016-year_emp 재직연수
from professor
where year_emp = (
    select max(year_emp)
    from professor
);
```



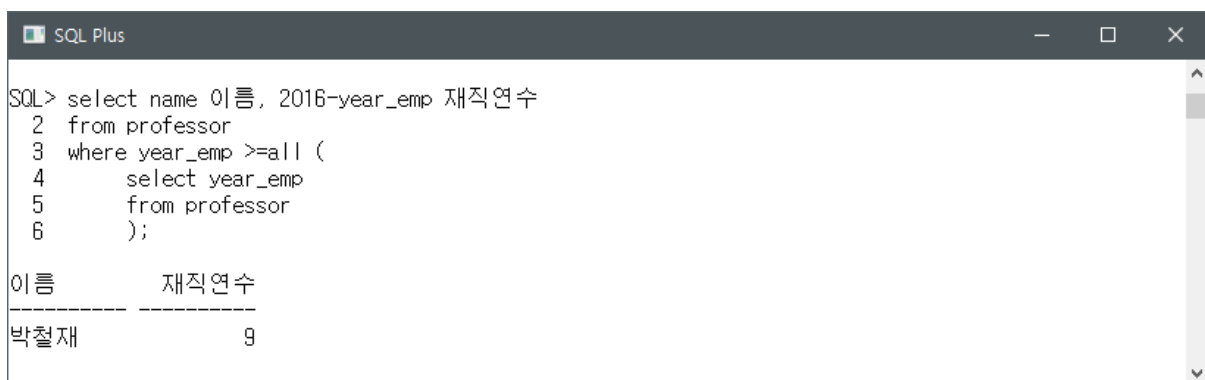
The screenshot shows a SQL Plus window titled '선택 SQL Plus'. The SQL prompt shows the query from the previous block. The result is displayed as a table with two columns: '이름' (Name) and '재직연수' (Tenure). The result shows '박철재' with a tenure of 9.

이름	재직연수
박철재	9

<그림 1-7(1)> 문제 1-7(1)의 SQL 문과 그 결과

(2) 중첩 질의 사용

```
select name 이름, 2016-year_emp 재직연수
from professor
where year_emp >=all (
    select year_emp
    from professor
);
```



The screenshot shows a SQL Plus window titled 'SQL Plus'. The SQL prompt shows the query from the previous block. The result is displayed as a table with two columns: '이름' (Name) and '재직연수' (Tenure). The result shows '박철재' with a tenure of 9.

이름	재직연수
박철재	9

<그림 1-7(2)> 문제 1-7(2)의 SQL 문과 그 결과

문제1-8. 같은 학과, 같은 주소를 갖는 학생 이름의 쌍을 찾아라. (단 동일 이름이 두 번 나와서는 안되며, 같은 쌍이 두 번 나와서도 안됨.)

```
select s1.name 학생 1, s2.name 학생 2
from student s1, student s2
where s1.stu_id < s2.stu_id
and s1.dept_id = s2.dept_id and s1.address = s2.address;
```



The screenshot shows a SQL Plus window titled '선택 SQL Plus'. The SQL prompt is followed by the query:

SQL> select s1.name 학생1, s2.name 학생2

2 from student s1, student s2

3 where s1.stu_id < s2.stu_id

4 and s1.dept_id = s2.dept_id

5 and s1.address = s2.address;

The results are displayed in a table with two columns: '학생1' and '학생2'. The data rows are:

김광식 김정현

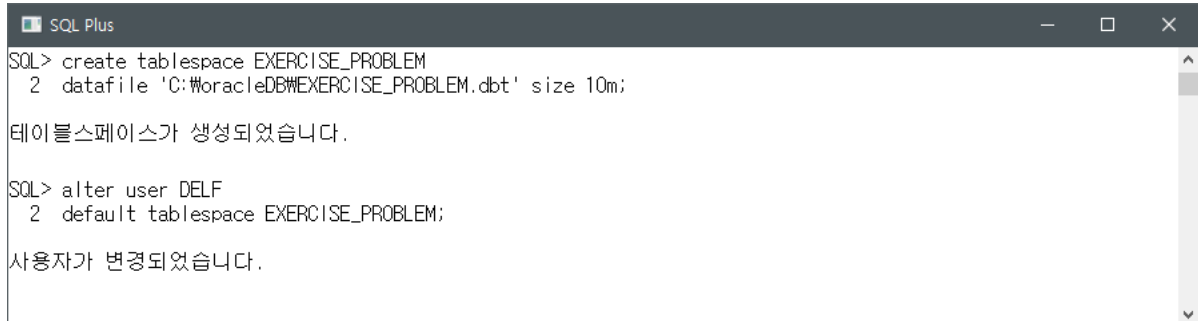
김정현 이상훈

김광식 이상훈

<그림 1-8> 문제 1-8 의 SQL 문과 그 결과

<문제 2>

- 연습문제 풀이를 위하여 새로운 TABLESPACE를 생성하고, USER의 DEFAULT TABLESPACE 변경



```

SQL> create tablespace EXERCISE_PROBLEM
  2  datafile 'C:\oracleDB\EXERCISE_PROBLEM.dbf' size 10m;

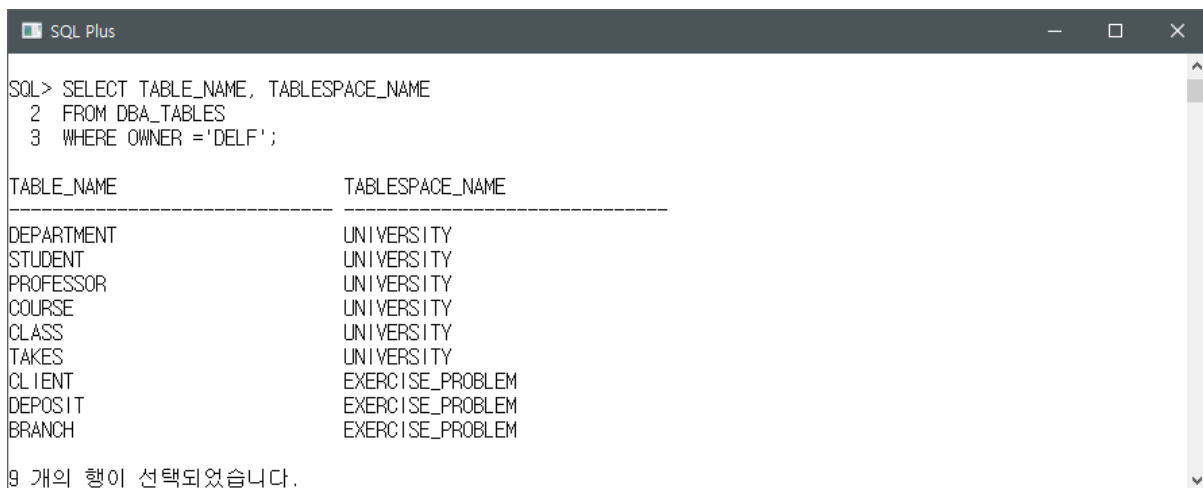
테이블스페이스가 생성되었습니다.

SQL> alter user DELF
  2  default tablespace EXERCISE_PROBLEM;

사용자가 변경되었습니다.
  
```

<그림2-A> USER의 DEFAULT TABLESPACE 변경

- 권한을 가진 USER **DELF**로 TABLESPACE **EXERCISE_PROBLEM**에 테이블을 생성한다.
생성한 테이블들은 아래와 같다.



```

SQL> SELECT TABLE_NAME, TABLESPACE_NAME
  2  FROM DBA_TABLES
  3  WHERE OWNER = 'DELF';
  
```

TABLE_NAME	TABLESPACE_NAME
DEPARTMENT	UNIVERSITY
STUDENT	UNIVERSITY
PROFESSOR	UNIVERSITY
COURSE	UNIVERSITY
CLASS	UNIVERSITY
TAKES	UNIVERSITY
CLIENT	EXERCISE_PROBLEM
DEPOSIT	EXERCISE_PROBLEM
BRANCH	EXERCISE_PROBLEM

9 개의 행이 선택되었습니다.

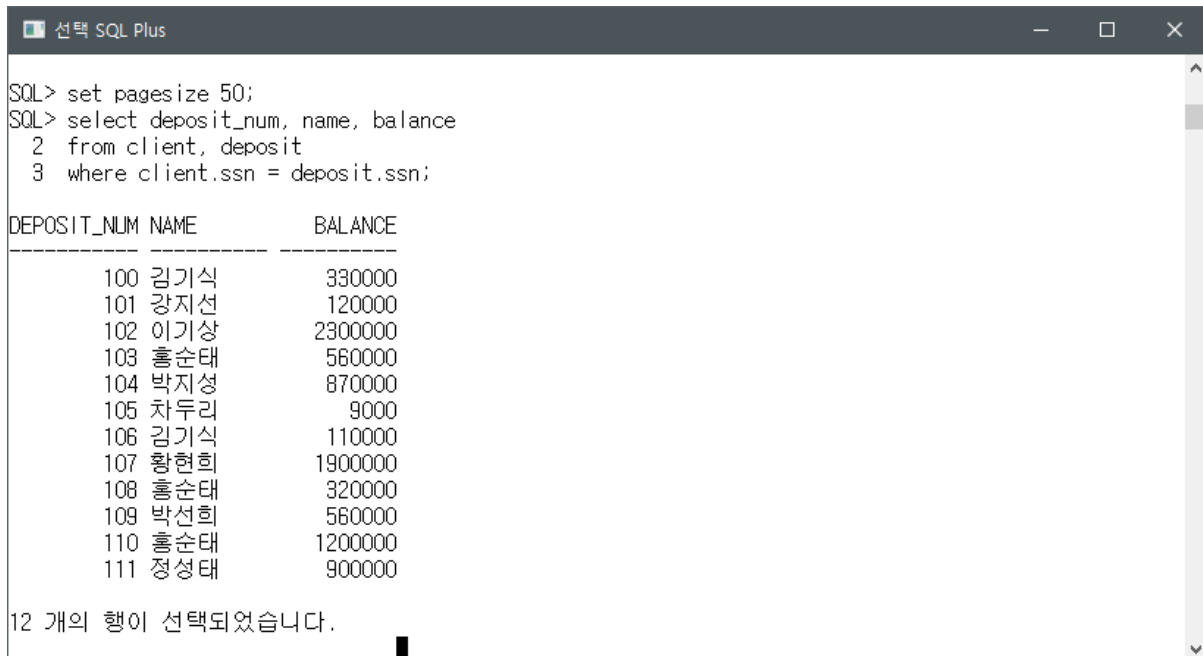
<그림2-B> DELF가 생성한 TABLE 목록

문제2-1. 모든 고객의 계좌번호, 이름, 그리고 예금 잔액을 검사하라.

select deposit_num, name, balance

from client, deposit

where client.ssn = deposit.ssn;



```

SQL> set pagesize 50;
SQL> select deposit_num, name, balance
  2  from client, deposit
  3  where client.ssn = deposit.ssn;

```

DEPOSIT_NUM	NAME	BALANCE
100	김기식	330000
101	강지선	120000
102	이기상	2300000
103	홍순태	560000
104	박지성	870000
105	차두리	9000
106	김기식	110000
107	황현희	1900000
108	홍순태	320000
109	박선희	560000
110	홍순태	1200000
111	정성태	900000

12 개의 행이 선택되었습니다.

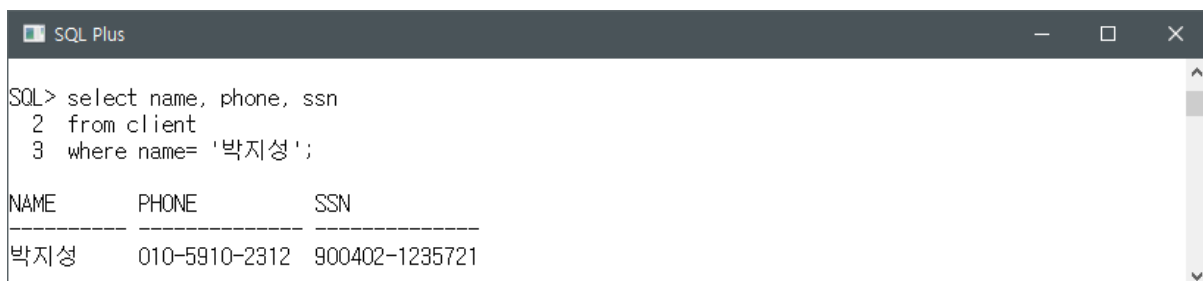
<그림2-1> 문제2-1의 SQL문과 그 결과

문제2-2. 이름이 '박지성'인 고객의 전화번호와 주민등록번호를 검색하라.

select name, phone, ssn

from client

where name= '박지성';



```

SQL> select name, phone, ssn
  2  from client
  3  where name= '박지성';

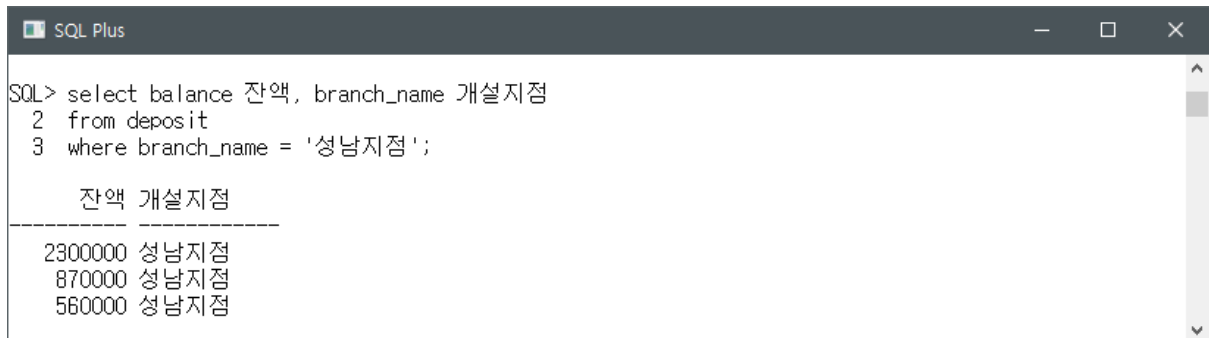
```

NAME	PHONE	SSN
박지성	010-5910-2312	900402-1235721

<그림2-2> 문제2-2의 SQL문과 그 결과

문제2-3. 지점 이름이 '성남지점'인 지점을 통해 개설된 모든 예금의 잔액을 검사하라.

select balance 잔액, branch_name 개설지점
from deposit
where branch_name = '성남지점';



SQL Plus window showing the following SQL query and results:

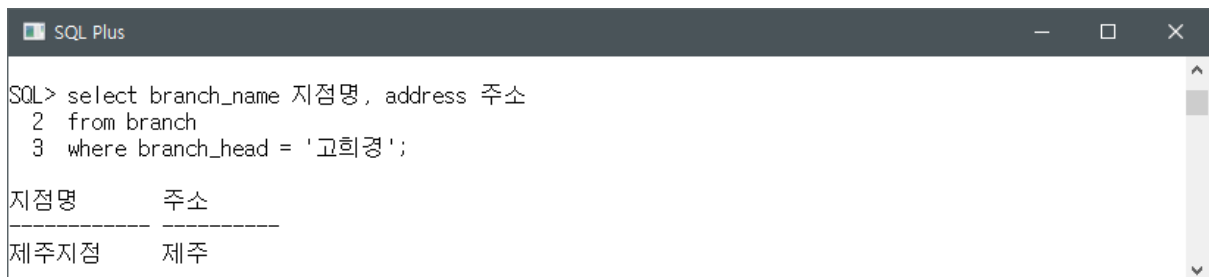
```
SQL> select balance 잔액, branch_name 개설지점
2  from deposit
3  where branch_name = '성남지점';
```

잔액	개설지점
2300000	성남지점
870000	성남지점
560000	성남지점

<그림2-3> 문제2-3의 SQL문과 그 결과

문제2-4. 지점장 이름이 '고희경'인 지점의 이름과 주소를 검색하라

select branch_name 지점명, address 주소
from branch
where branch_head = '고희경';



SQL Plus window showing the following SQL query and results:

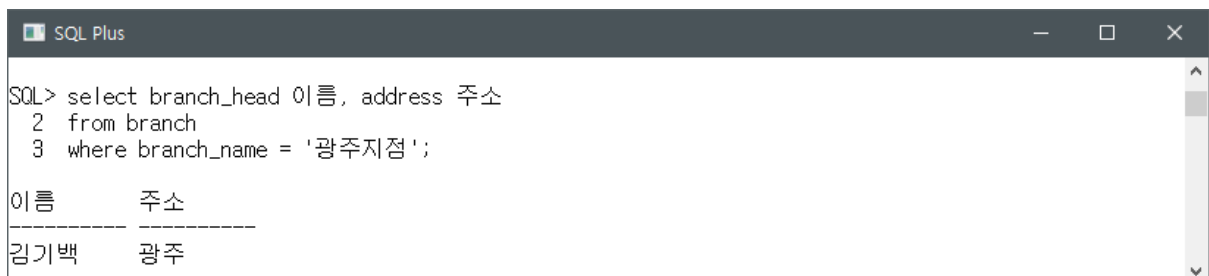
```
SQL> select branch_name 지점명, address 주소
2  from branch
3  where branch_head = '고희경';
```

지점명	주소
제주지점	제주

<그림2-4> 문제2-4의 SQL문과 그 결과

문제2-5. 지점 이름이 '광주지점'인 지점의 지점장 이름과 주소를 검색하라.

select branch_head 이름, address 주소
from branch
where branch_name = '광주지점';



SQL Plus window showing the following SQL query and results:

```
SQL> select branch_head 이름, address 주소
2  from branch
3  where branch_name = '광주지점';
```

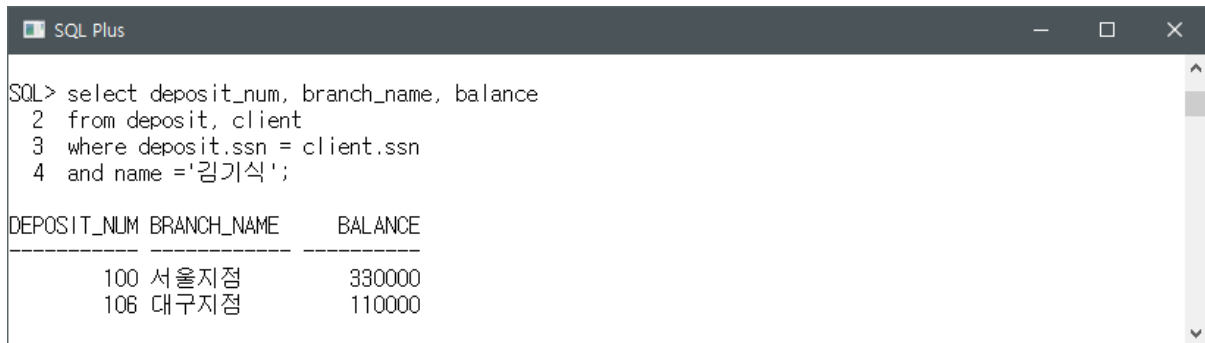
이름	주소
김기백	광주

<그림2-5> 문제2-5의 SQL문과 그 결과

문제2-6. 이름이 '김기식'인 고객이 소유한 예금의 계좌번호, 개설지점의 이름, 잔액을 검색하라.

(1) 카티션 테이블

```
select deposit_num, branch_name, balance
from deposit, client
where deposit.ssn = client.ssn
and name = '김기식';
```



SQL Plus window showing the following SQL query and results:

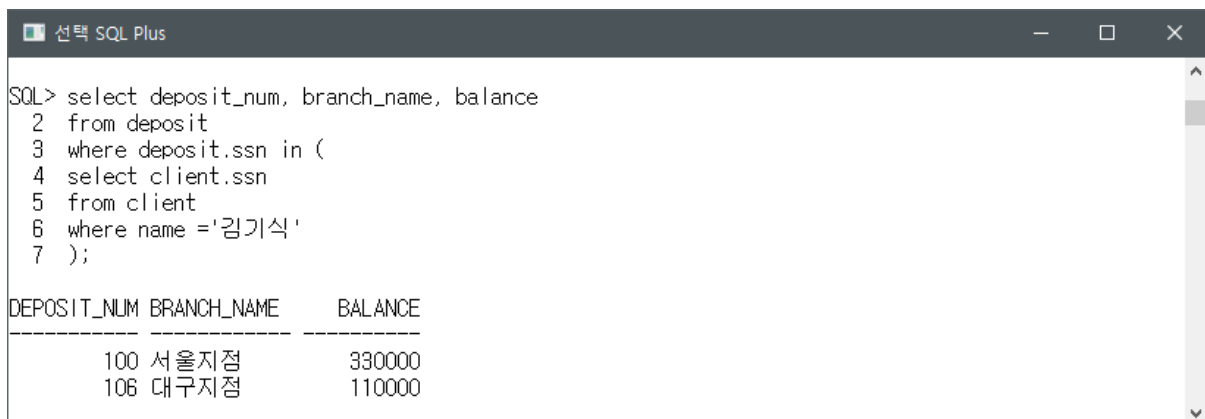
```
SQL> select deposit_num, branch_name, balance
2  from deposit, client
3  where deposit.ssn = client.ssn
4  and name = '김기식';
```

DEPOSIT_NUM	BRANCH_NAME	BALANCE
100	서울지점	330000
106	대구지점	110000

<그림2-6(1)> 문제2-6(1)의 SQL문과 그 결과

(2) 중첩 질의 사용

```
select deposit_num, branch_name, balance
from deposit
where deposit.ssn in (
    select client.ssn
    from client
    where name = '김기식'
);
```



선택 SQL Plus window showing the following nested query and results:

```
SQL> select deposit_num, branch_name, balance
2  from deposit
3  where deposit.ssn in (
4  select client.ssn
5  from client
6  where name = '김기식'
7  );
```

DEPOSIT_NUM	BRANCH_NAME	BALANCE
100	서울지점	330000
106	대구지점	110000

<그림2-6(2)> 문제2-6(2)의 SQL문과 그 결과

문제2-7. '성남지점'에서 계좌를 개설한 고객의 이름과 주소, 그리고 예금 잔액을 검색하라

```
select name, address, balance
from client, deposit
where client.ssn = deposit.ssn and branch_name = '성남지점';
```



```

SQL> select name, address, balance
2  from client, deposit
3  where client.ssn = deposit.ssn
4  and branch_name = '성남지점';

```

NAME	ADDRESS	BALANCE
이기상	대전	2300000
박지성	서울	870000
박선희	서울	560000

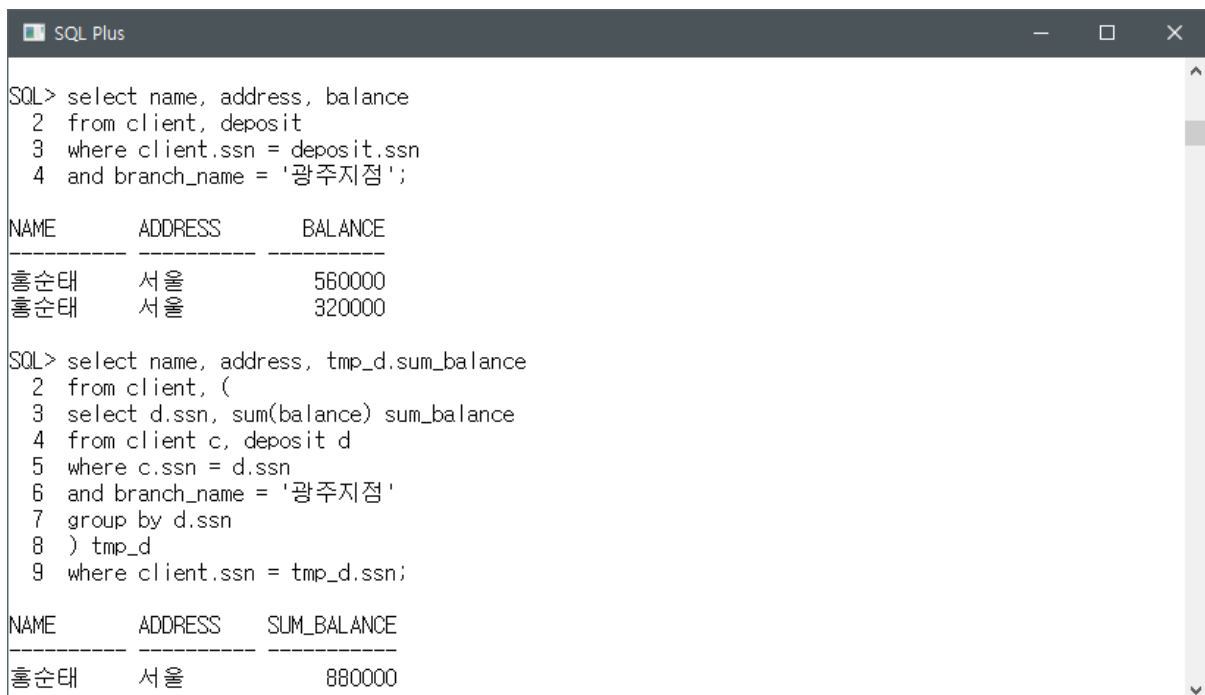
<그림2-7(1)> 문제2-7의 SQL문과 그 결과

- ✓ 위의 SQL Query에서는 같은 사람이 같은 지점에서 2개 이상의 계좌 개설 시, 이름이 중복되어 출력되는데, 아래는 그를 보완한 SQL Query임. 같은 사람의 다른 계좌 존재 시, 잔액은 각 계좌의 합으로 출력함 (차이를 보이기 위해 '성남지점'이 아닌 '광주지점'으로 예를 들)

```

select name, address, tmp_d.sum_balance
from client, (
    select d.ssn, sum(balance) sum_balance
    from client c, deposit d
    where c.ssn = d.ssn and branch_name = '광주지점'
    group by d.ssn
) tmp_d
where client.ssn = tmp_d.ssn;

```



```

SQL> select name, address, balance
2  from client, deposit
3  where client.ssn = deposit.ssn
4  and branch_name = '광주지점';

```

NAME	ADDRESS	BALANCE
홍순태	서울	560000
홍순태	서울	320000

```

SQL> select name, address, tmp_d.sum_balance
2  from client, (
3  select d.ssn, sum(balance) sum_balance
4  from client c, deposit d
5  where c.ssn = d.ssn
6  and branch_name = '광주지점'
7  group by d.ssn
8  ) tmp_d
9  where client.ssn = tmp_d.ssn;

```

NAME	ADDRESS	SUM_BALANCE
홍순태	서울	880000

<그림2-7(2)> 문제2-7의 Query 비교(위: 개선 전, 아래: 개선 후)

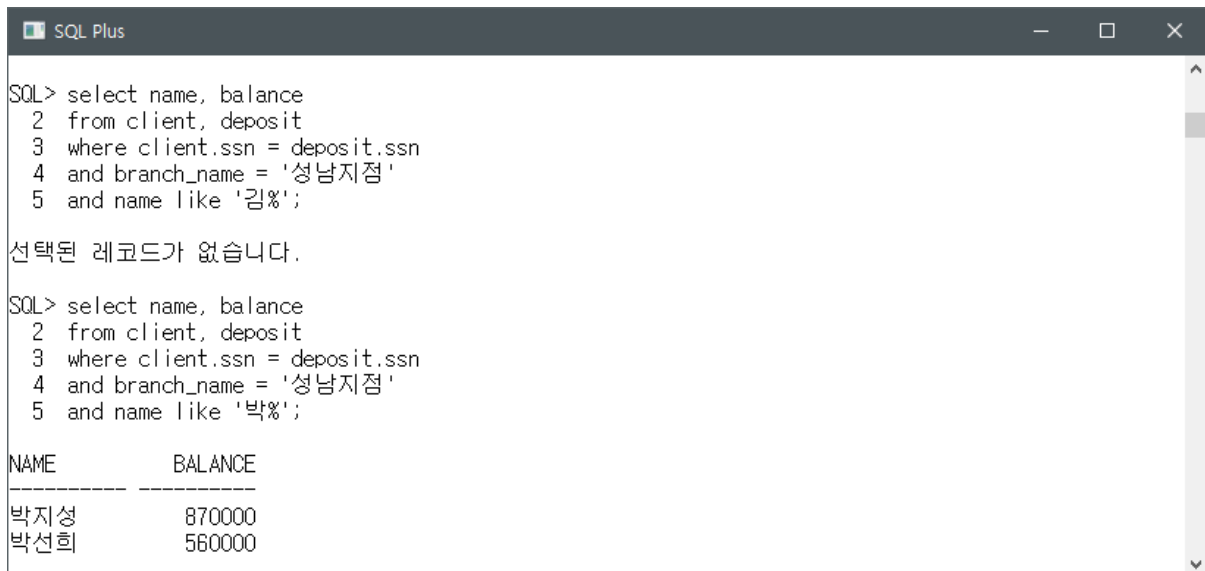
문제2-8. '성남지점'에서 계좌를 개설한 고객 중 김씨 성을 가진 고객의 이름과 예금 잔액을 검색하라.

select name, balance

from client, deposit

where client.ssn = deposit.ssn **and** branch_name = '성남지점' **and** name like '김%';

- ✓ '김씨'로 검색하여 해당되는 레코드가 존재하지 않아, 해당 Query를 테스트하기 위하여 '박씨'로도 검색하여 봄.



```

SQL> select name, balance
  2  from client, deposit
  3  where client.ssn = deposit.ssn
  4  and branch_name = '성남지점'
  5  and name like '김%';

선택된 레코드가 없습니다.

SQL> select name, balance
  2  from client, deposit
  3  where client.ssn = deposit.ssn
  4  and branch_name = '성남지점'
  5  and name like '박%';

NAME          BALANCE
-----
박지성         870000
박선희         560000
  
```

<그림2-8> 문제2-8의 SQL문과 그 결과

문제2-9. 예금 잔액이 10만원 이상인 계좌를 소유한 고객의 이름을 검색하라.

select distinct name 이름

from client, deposit

where client.ssn = deposit.ssn **and** balance >= 100000;



```

SQL> select distinct name 이름
  2  from client, deposit
  3  where client.ssn = deposit.ssn
  4  and balance >= 100000;

이름
-----
황현희
홍순태
박지성
정성태
강지선
이기상
박선희
김기식

8 개의 행이 선택되었습니다.
  
```

<그림2-9> 문제2-9의 SQL문과 그 결과

문제2-10. 예금 잔액이 10만원 이상인 계좌가 개설된 지점의 이름과 지점장 이름을 검색하라.

select distinct branch.branch_name 지점명, branch_head 지점장
from deposit, branch
where deposit.branch_name = branch.branch_name **and** balance >= 100000;

```

SQL> select distinct branch.branch_name 지점명, branch_head 지점장
2  from deposit, branch
3  where deposit.branch_name = branch.branch_name
4  and balance >= 100000;

```

지점명	지점장
대구지점	김기식
성남지점	박찬주
대전지점	이연희
제주지점	고희경
광주지점	김기백
서울지점	강동희

6 개의 행이 선택되었습니다.

<그림2-10> 문제2-10의 SQL문과 그 결과

문제2-11. 예금을 개설한 지점의 지점장과 이름이 같은 고객이 소유한 예금의 계좌번호 잔액, 그리고 개설지점 이름을 검색하라.

select deposit_num, balance, b.branch_name
from client c, deposit d, branch b
where c.ssn = d.ssn and d.branch_name = b.branch_name **and** c.name = b.branch_head;

```

SQL> select deposit_num, balance, b.branch_name
2  from client c, deposit d, branch b
3  where c.ssn=d.ssn and d.branch_name = b.branch_name
4  and c.name = b.branch_head;

```

DEPOSIT_NUM	BALANCE	BRANCH_NAME
106	110000	대구지점

<그림2-11> 문제2-11의 SQL문과 그 결과

문제2-12. '서울지점'에서 계좌를 개설한 고객들 중에서 남자 고객의 이름과 예금 잔액을 검색하라.

select name, balance
from client c, deposit d
where c.ssn = d.ssn **and** d.branch_name = '서울지점' **and** c.ssn like '____-1%';



SQL> select name, balance
2 from client c, deposit d
3 where c.ssn = d.ssn
4 and d.branch_name = '서울지점'
5 and c.ssn like '____-1%';

NAME	BALANCE
김기식	330000
홍순태	1200000

<그림2-12> 문제2-12의 SQL문과 그 결과

문제2-13. 주민등록번호상의 생일이 3월인 모든 고객의 이름과 소유한 예금의 계좌번호를 검색하라.

select name, deposit_num
from client c, deposit d
where c.ssn = d.ssn **and** c.ssn like '___3%';



2 from client c, deposit d
3 where c.ssn = d.ssn
4 and c.ssn like '___3%';

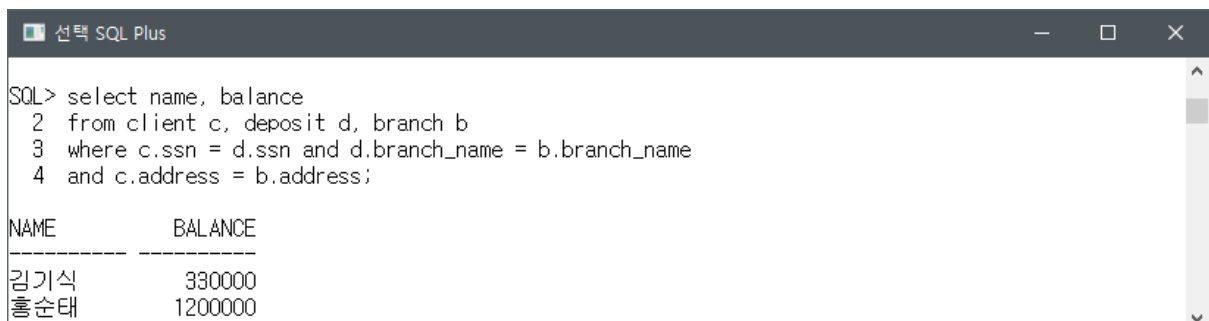
NAME	DEPOSIT_NUM
홍순태	103
홍순태	108
홍순태	110
정성태	111

SQL>

<그림2-13> 문제2-13의 SQL문과 그 결과

문제2-14. 자신의 주소와 같은 지점에 계좌를 소유하고 있는 고객의 이름과 예금 잔액을 검색하라

select name, balance
from client c, deposit d, branch b
where c.ssn = d.ssn **and** d.branch_name = b.branch_name **and** c.address = b.address;



선택 SQL Plus

SQL> select name, balance
2 from client c, deposit d, branch b
3 where c.ssn = d.ssn and d.branch_name = b.branch_name
4 and c.address = b.address;

NAME	BALANCE
김기식	330000
홍순태	1200000

<그림2-14> 문제2-14의 SQL문과 그 결과

문제2-15. '성남지점'과 거래하고 있는 고객의 숫자를 검색하라.

```
select count(distinct ssn)
from deposit
where branch_name = '성남지점';
```



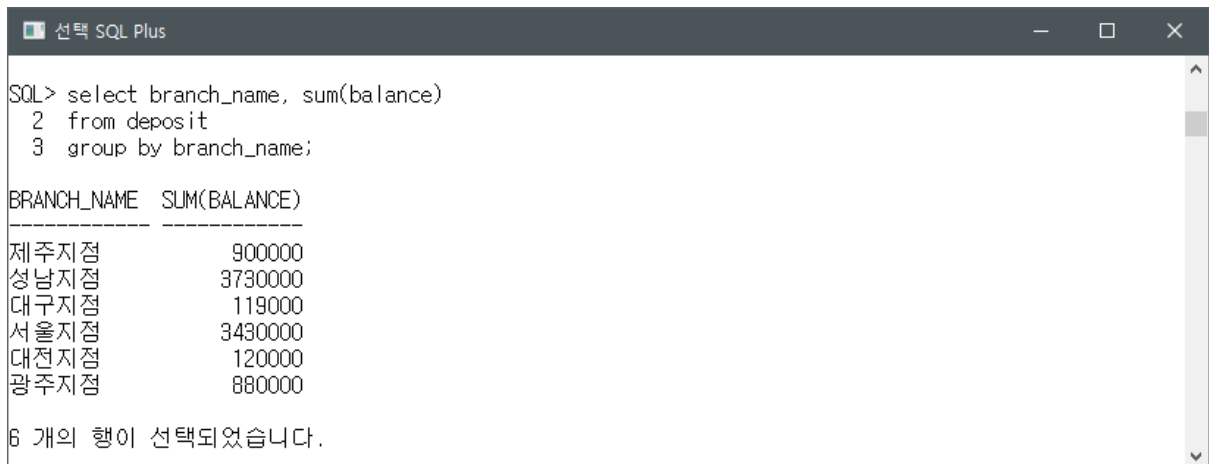
SQL Plus window showing the SQL query and its result. The query is: select count(distinct ssn) from deposit where branch_name = '성남지점';. The result is: COUNT(DISTINCTSSN) 3.

COUNT(DISTINCTSSN)
3

<그림2-15> 문제2-15의 SQL문과 그 결과

문제2-16. 각 지점 별 잔액의 총합을 검색하라.

```
select branch_name, sum(balance)
from deposit
group by branch_name;
```



SQL Plus window showing the SQL query and its result. The query is: select branch_name, sum(balance) from deposit group by branch_name;. The result is a table with 6 rows and 2 columns: BRANCH_NAME and SUM(BALANCE). The rows are: 제주지점 (900000), 성남지점 (3730000), 대구지점 (119000), 서울지점 (3430000), 대전지점 (120000), 광주지점 (880000). Below the table, it says: 6 개의 행이 선택되었습니다.

BRANCH_NAME	SUM(BALANCE)
제주지점	900000
성남지점	3730000
대구지점	119000
서울지점	3430000
대전지점	120000
광주지점	880000

6 개의 행이 선택되었습니다.

<그림2-16> 문제2-16의 SQL문과 그 결과

문제2-17. 고객 이름 별 예금 잔액의 총합을 검색하라.

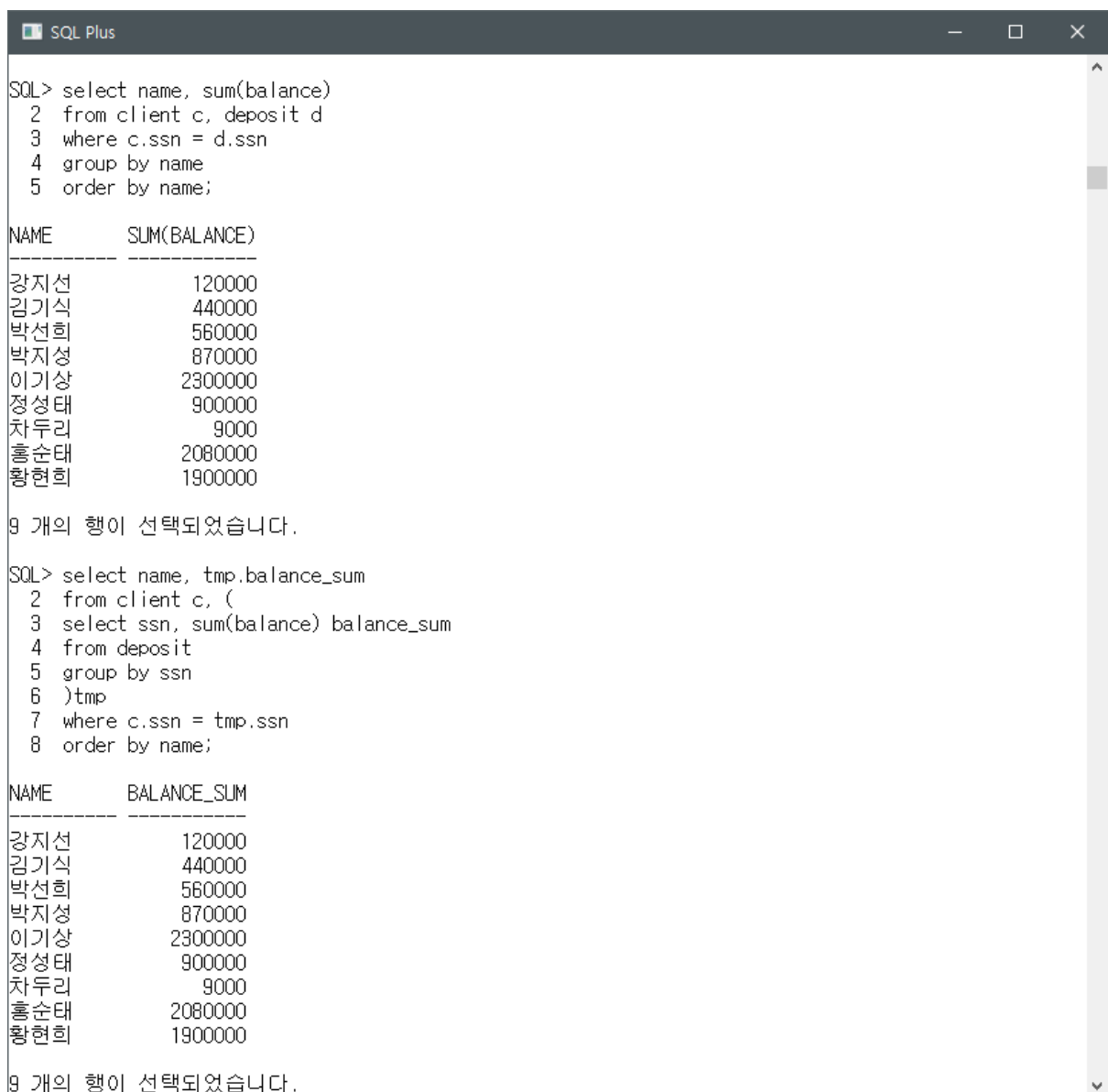
```
select name, sum(balance)
from client c, deposit d
where c.ssn = d.ssn
group by name
order by name;
```

- ✓ 위의 Query는 현재 다루고 있는 테이블에서는 동명이인이 존재하지 않아 올바른 결과가 출력되나, 동명이인의 계좌는 하나로 계산 되는 오류가 생김. 아래는 개선 된 Query.

```

select name, tmp.balance_sum
from client c, (
    select ssn, sum(balance) balance_sum
    from deposit
    group by ssn
) tmp
where c.ssn = tmp.ssn
order by name;

```



SQL Plus window showing the execution of two SQL queries and their results.

Query 1:

```

SQL> select name, sum(balance)
2  from client c, deposit d
3  where c.ssn = d.ssn
4  group by name
5  order by name;

```

NAME	SUM(BALANCE)
강지선	120000
김기식	440000
박선희	560000
박지성	870000
이기상	2300000
정성태	900000
차두리	9000
홍순태	2080000
황현희	1900000

9 개의 행이 선택되었습니다.

Query 2:

```

SQL> select name, tmp.balance_sum
2  from client c, (
3  select ssn, sum(balance) balance_sum
4  from deposit
5  group by ssn
6  )tmp
7  where c.ssn = tmp.ssn
8  order by name;

```

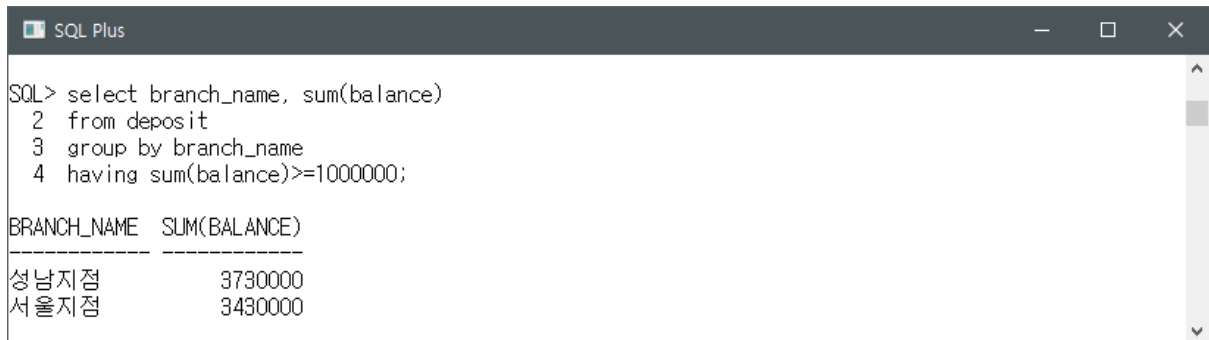
NAME	BALANCE_SUM
강지선	120000
김기식	440000
박선희	560000
박지성	870000
이기상	2300000
정성태	900000
차두리	9000
홍순태	2080000
황현희	1900000

9 개의 행이 선택되었습니다.

<그림2-17> 문제2-17의 SQL문과 그 결과 (위: 개선 전, 아래: 개선 후)

문제2-18. 잔액의 합이 100만원 이상인 지점 이름과 잔액의 합을 검색하라.

```
select branch_name, sum(balance)
from deposit
group by branch_name
having sum(balance) >= 1000000;
```



SQL Plus window showing the SQL query and its results. The query is: select branch_name, sum(balance) from deposit group by branch_name having sum(balance) >= 1000000; The results show two branches: 성남지점 with a sum of 3730000 and 서울지점 with a sum of 3430000.

BRANCH_NAME	SUM(BALANCE)
성남지점	3730000
서울지점	3430000

<그림2-18> 문제2-18의 SQL문과 그 결과

문제2-19. 지점별로 예금 잔액이 100만원 이상인 고객의 숫자를 검색하라.

```
select branch_name, count(*)
from deposit
where balance >= 1000000
group by branch_name;
```



SQL Plus window showing the SQL query and its results. The query is: select branch_name, count(*) from deposit where balance >= 1000000 group by branch_name; The results show two branches: 성남지점 with a count of 1 and 서울지점 with a count of 2.

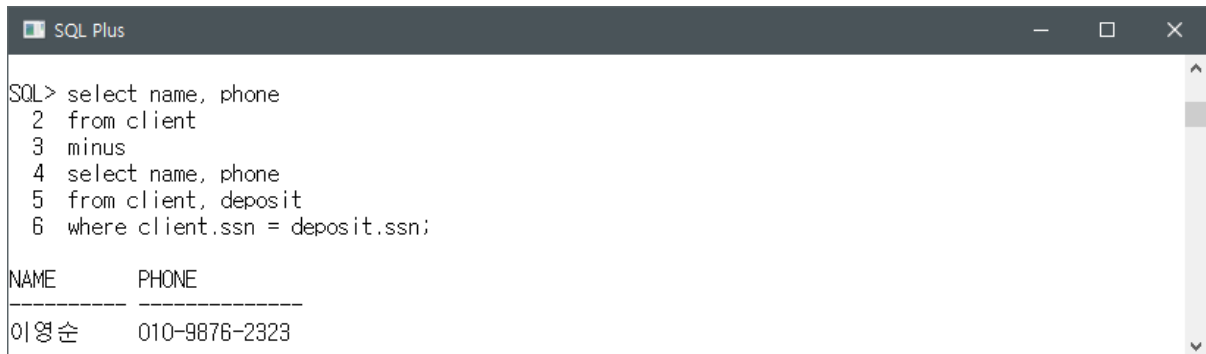
BRANCH_NAME	COUNT(*)
성남지점	1
서울지점	2

<그림2-19> 문제2-19의 SQL문과 그 결과

문제2-20. 예금 계좌를 소유하고 있지 않은 고객의 이름과 전화번호를 검색하라.

(1) 집합연산 minus 사용

```
select name, phone
from client
minus
select name, phone
from client, deposit
where client.ssn = deposit.ssn;
```



SQL Plus window showing the SQL query and its result. The query uses the MINUS operator to find clients who do not have a deposit account. The result shows one client: 이영순 with phone number 010-9876-2323.

NAME	PHONE
이영순	010-9876-2323

<그림2-20(1)> 문제2-20(1)의 SQL문과 그 결과

(2) 중첩 질의 사용

```
select name, phone
from client
where ssn not in (
select ssn
from deposit
);
```



SQL Plus window showing the SQL query and its result. The query uses a subquery to find clients whose SSN is not in the deposit table. The result shows one client: 이영순 with phone number 010-9876-2323.

NAME	PHONE
이영순	010-9876-2323

<그림2-20(2)> 문제2-20(2)의 SQL문과 그 결과