

Contents

I.	DDL로 정의한 Schema	2
II.	INSERT 이후의 Database	4
III.	Query 적용예제	7

I. DDL로 정의한 Schema

: Data Definition Language를 이용하여 Schema를 정의한 과정을 보이시오.

MySQL을 설치할 시 같이 설치되는 MySQL Workbench를 이용하면 아주 쉽게 스키마를 정의할 수 있습니다. 빈칸을 채우고, 체크박스를 누르는 것만으로 자동으로 DDL로 변환하여 Data base에 Apply 할 수 있게 해 줍니다.

다음은 'branch' table에 대해 실제 적용해 본 예입니다. '그림 1.'에 보이는 것 처럼, 원하는 설정대로 정의 후, Apply 버튼을 누르면 정의한 Schema에 대해 DDL로 변환된 팝업되어 나타납니다.

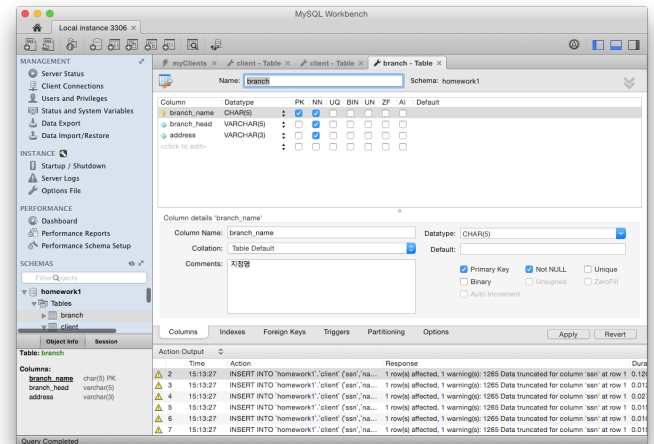


그림 1. Schema 설정 예시

Schema를 정의함에 있어, 때로는 다른 Schema로 부터

Foreign Key를 가져와야 하는 경우가 있습니다.

MySQL Work bench는 이런 상황에 대해서도 아주 간편하게 처리할 수 있는 기능을 제공합니다.

'deposit' table은 이런 상황을 아주 잘 보여주는 좋은 예시입니다. 우선은 '그림 1.'에서 처럼 정의하고자 하는 모든 Attribute들을 정의 해줍니다. 그 후, foreign key에 해당하는 Attribute에 대해 '그림 2.'에서 처럼 설정해주면 손쉽게 FK를 정의할 수 있게 됩니다.

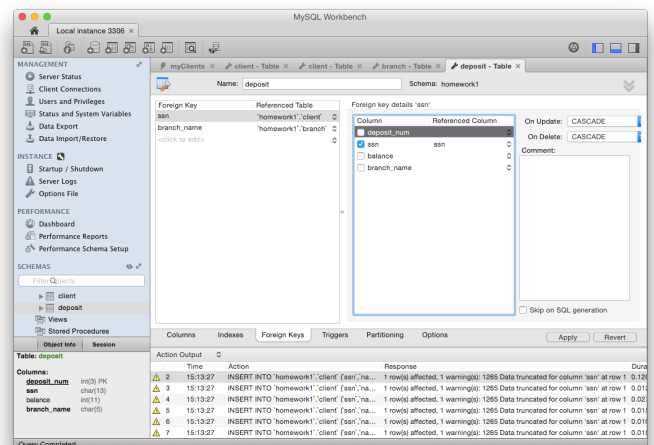


그림 2. FK 정의 예시

이러한 정의/설정 과정도 결국은 인간의 일인지라, 때로는 정의한 Attribute에 잘못된 데이터타입을 기입하는 경우도 있습니다.

MySQL Workbench는 이런 상황에 대해서도 손쉽게 대처할 수 있는 방안을 제공하고 있습니다.

'그림 3.'의 'client' table은 '주민등록번호는 13자리이다.'라는 사실만 생각해버린 결과, 가운데 '-'가 있다는 사실을 망각하여, 주민등록번호의 데이터 타입을 잘못 정의할 수 있는 좋은 예입니다. 해당 데이터 타입에 대해 14자리로 바꾸어 주어야 하는 상황에 빠지게 되고, 이에 대해선, 우클릭 하면 등장하는 'Alter Table...' 메뉴로 접근하면 됩니다. 수정과정은 '그림 1.'과 같습니다.

하지만 주민등록번호는 이미 '그림 2.'에서 FK로 사용한 사실이 있는 바, MySQL Workbench는 아쉽게도 이에 대해선 오류를 출력할 뿐, 수정을 허락하지 않습니다. 따라서 이에 대해선, 해당 Attribute에 대한 FK를 해제한 후 수정해야 합니다.

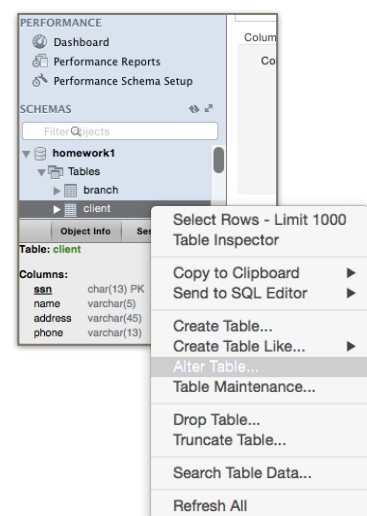


그림 3. Column 수정 예시

다음은 MySQL Workbench에 의해 자동으로 DDL로 변환된 Schema 정의문입니다. 잘 된 경우에는 문제가 없지만, 안된 경우에는 그림 6. 에서 처럼 오류의 원인을 알려줍니다. DBA는 이러한 오류 출력문을 보고, 오류에 대한 적절한 대처를 해야 할 것 입니다.

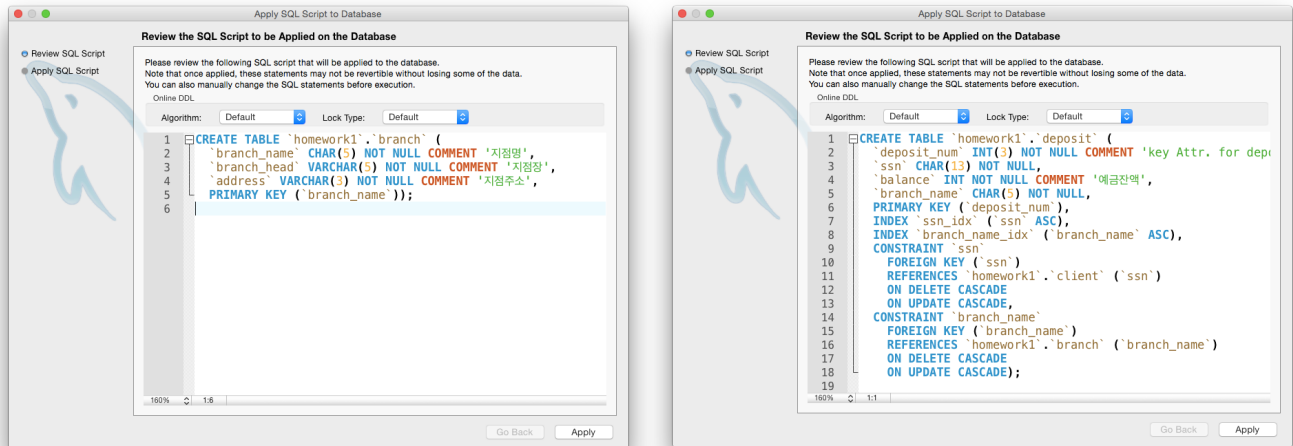


그림 4. 'branch'와 'deposit'의 정의

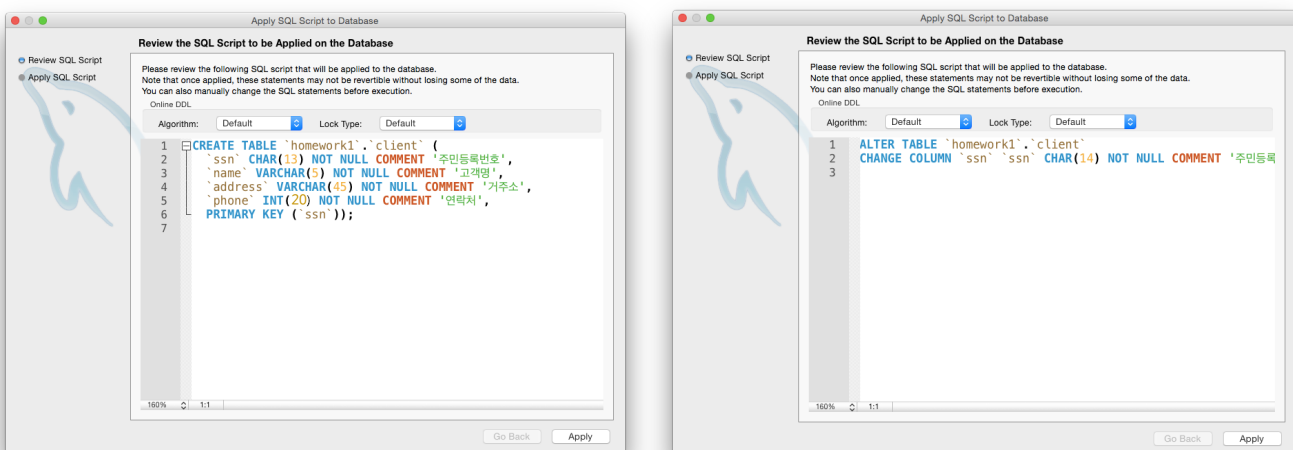


그림 5. 'Client' 정의와 수정 예시

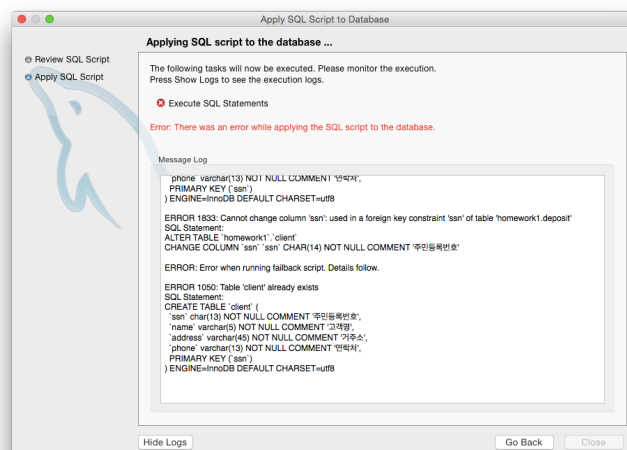


그림 6. 수정불가 오류 예시

II. INSERT 이후의 Database

: 생성한 테이블 각각에 정보를 INSERT 하고, 이후의 Database 상태를 보이시오.

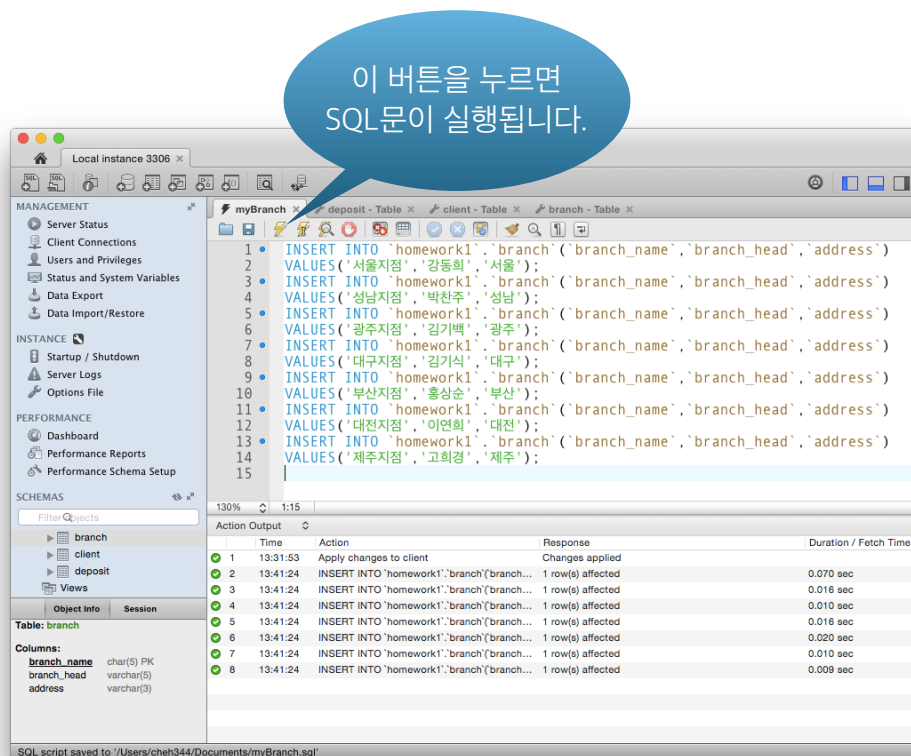


그림 7. 'branch' table에 INSERT 를 진행

```
INSERT INTO `homework1`.`branch` (`branch_name`,`branch_head`,`address`)
VALUES ('서울지점','강동희','서울');
```

```
INSERT INTO `homework1`.`branch` (`branch_name`,`branch_head`,`address`)
VALUES ('성남지점','박찬주','성남');
```

```
INSERT INTO `homework1`.`branch` (`branch_name`,`branch_head`,`address`)
VALUES ('광주지점','김기백','광주');
```

```
INSERT INTO `homework1`.`branch` (`branch_name`,`branch_head`,`address`)
VALUES ('대구지점','김기식','대구');
```

```
INSERT INTO `homework1`.`branch` (`branch_name`,`branch_head`,`address`)
VALUES ('부산지점','홍상순','부산');
```

```
INSERT INTO `homework1`.`branch` (`branch_name`,`branch_head`,`address`)
VALUES ('대전지점','이연희','대전');
```

```
INSERT INTO `homework1`.`branch` (`branch_name`,`branch_head`,`address`)
VALUES ('제주지점','고희경','제주');
```

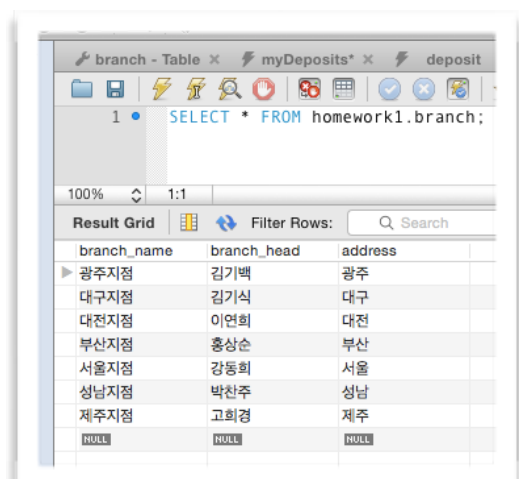


그림 8. 'branch' table 출력

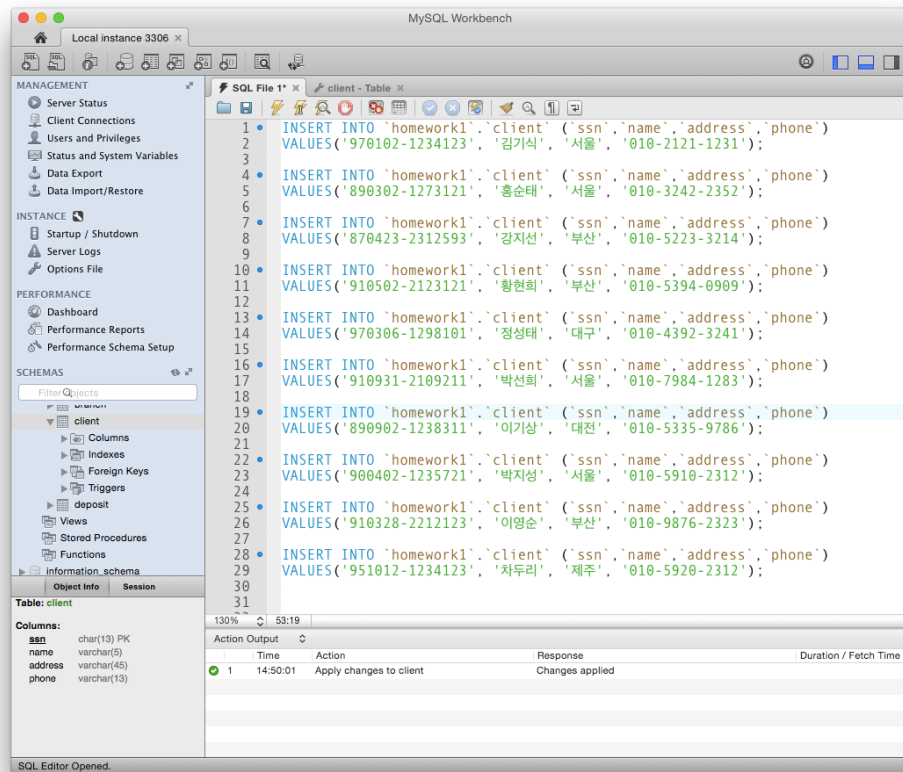


그림 7. 'client' table에 INSERT 를 진행

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('970102-1234123', '김기식', '서울', '010-2121-1231');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('890302-1273121', '홍순태', '서울', '010-3242-2352');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('870423-2312593', '강지선', '부산', '010-5223-3214');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('910502-2123121', '황현희', '부산', '010-5394-0909');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('970306-1298101', '정성태', '대구', '010-4392-3241');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('910931-2109211', '박선희', '서울', '010-7984-1283');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('890902-1238311', '이기상', '대전', '010-5335-9786');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('900402-1235721', '박지성', '서울', '010-5910-2312');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('910328-2212123', '이영순', '부산', '010-9876-2323');
```

```
INSERT INTO 'homework1'. 'client' ('ssn','name','address','phone')
VALUES('951012-1234123', '차두리', '제주', '010-5920-2312');
```

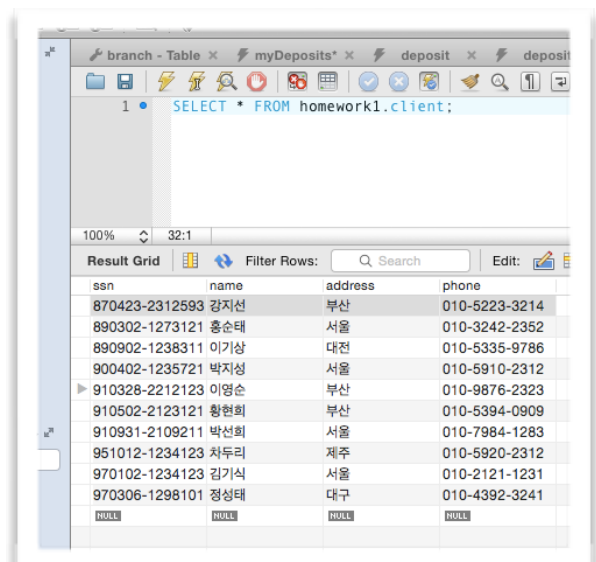


그림 8. 'client' table 출력

```

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES (100, (SELECT ssn FROM client WHERE ssn='970102-1234123'),
        330000, (SELECT branch_name FROM branch WHERE branch_name='서울지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES (101, (SELECT ssn FROM client WHERE ssn='870423-2312593'),
        120000, (SELECT branch_name FROM branch WHERE branch_name='대전지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 102, (SELECT ssn FROM client WHERE ssn='890902-1248311'),
        2300000, (SELECT branch_name FROM branch WHERE branch_name='성남지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 103, (SELECT ssn FROM client WHERE ssn='890302-1273121'),
        560000, (SELECT branch_name FROM branch WHERE branch_name='광주지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 104, (SELECT ssn FROM client WHERE ssn='900402-1235721'),
        870000, (SELECT branch_name FROM branch WHERE branch_name='성남지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 105, (SELECT ssn FROM client WHERE ssn='951012-1234123'),
        9000, (SELECT branch_name FROM branch WHERE branch_name='대구지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 106, (SELECT ssn FROM client WHERE ssn='970102-1234123'),
        110000, (SELECT branch_name FROM branch WHERE branch_name='대구지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 107, (SELECT ssn FROM client WHERE ssn='910502-2123121'),
        1900000, (SELECT branch_name FROM branch WHERE branch_name='서울지점'));

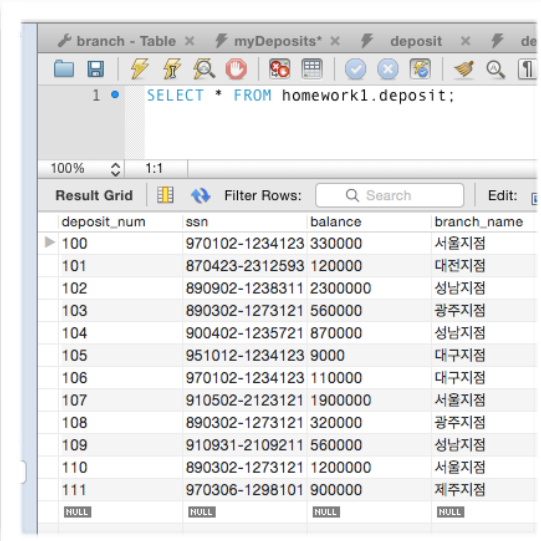
INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 108, (SELECT ssn FROM client WHERE ssn='890302-1273121'),
        320000, (SELECT branch_name FROM branch WHERE branch_name='광주지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 109, (SELECT ssn FROM client WHERE ssn='910931-2109211'),
        560000, (SELECT branch_name FROM branch WHERE branch_name='성남지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 110, (SELECT ssn FROM client WHERE ssn='890302-1273121'),
        1200000, (SELECT branch_name FROM branch WHERE branch_name='서울지점'));

INSERT INTO 'homework1'. 'deposit' ('deposit_num', 'ssn', 'balance', 'branch_name')
VALUES ( 111, (SELECT ssn FROM client WHERE ssn='970306-1298101'),
        900000, (SELECT branch_name FROM branch WHERE branch_name='제주지점'));

```



deposit_num	ssn	balance	branch_name
100	970102-1234123	330000	서울지점
101	870423-2312593	120000	대전지점
102	890902-1238311	2300000	성남지점
103	890302-1273121	560000	광주지점
104	900402-1235721	870000	성남지점
105	951012-1234123	9000	대구지점
106	970102-1234123	110000	대구지점
107	910502-2123121	1900000	서울지점
108	890302-1273121	320000	광주지점
109	910931-2109211	560000	성남지점
110	890302-1273121	1200000	서울지점
111	970306-1298101	900000	제주지점

그림 9. 'deposit' table 출력

III. Query 적용예제

: 생성한 스키마에 다음 Query를 적용하여보시오.

1. 모든 고객의 주민번호, 이름, 그리고 예금 잔액을 검색하라.
2. 이름이 '박지성'인 고객의 전화번호와 주민번호를 검색하라.
3. 지점 이름이 '성남지점'인 지점을 통해 개설된 모든 예금의 잔액을 검색하라.
4. 지점장 이름이 '고소영'인 지점의 이름과 주소를 검색하라. (삭제)
5. 지점 이름이 '광주지점'인 지점의 지점장 이름과 주소를 검색하라.
6. 이름이 '김광식' '김기식'인 고객이 소유한 예금의 개설지점의 이름, 잔액을 검색하라. (수정)
7. '성남지점'에서 예금이 있는 고객의 이름과 주소, 그리고 예금 잔액을 검색하라.
8. '성남지점'에서 예금이 있는 고객 중 김씨 박씨 성을 가진 고객의 이름과 예금 잔액을 검색하라. (수정)
9. 10만원 이상의 예금이 있는 고객의 이름을 검색하라.
10. 10만원 이상의 예금이 있는 고객을 가진 지점의 이름과 지점장 이름을 검색하라.
11. 예금을 개설한 지점의 지점장과 이름이 같은 고객이 소유한 예금의 잔액, 개설지점 이름을 검색하라.
12. '서울지점'에서 계좌를 개설한 고객들 중에서 남자 고객의 이름과 예금 잔액을 검색하라. (삭제)
13. 주민등록번호 상의 생일이 3월인 모든 고객의 이름과 소유한 예금의 잔액을 검색하라.
14. 자신의 주소와 같은 지점에 계좌를 소유하고 있는 고객의 이름과 예금 잔액을 검색하라.
15. '성남지점'과 거래하고 있는 고객의 숫자를 검색하라.
16. 각 지점별 잔액의 총합을 검색하라.
17. 고객 이름별 예금 잔액의 총합을 검색하라.
18. 잔액의 합이 100만원 이상인 지점 이름과 잔액의 합을 검색하라.
19. 지점별로 예금 잔액이 100만원 이상인 고객의 숫자를 검색하라.
20. 예금 계좌를 소유하고 있지 않은 고객의 이름과 전화번호를 검색하라.

1. 모든 고객의 주민번호, 이름, 그리고 예금 잔액을 검색하라.

```
SELECT ssn, name, SUM(balance)
FROM client natural join deposit
GROUP BY ssn
```

ssn	name	SUM(balance)
870423-2312593	김지선	120000
890302-1273121	홍순태	2080000
890902-1238311	이기상	2300000
900402-1235721	박지성	870000
910502-2123121	황현희	1900000
910931-2109211	박선희	560000
951012-1234123	차두리	9000
970102-1234123	김기식	440000
970306-1298101	정성태	900000

2. 이름이 '박지성'인 고객의 전화번호와 주민번호를 검색하라.

```
SELECT c.name, c.phone, c.ssn
FROM client c
WHERE name='박지성'
```

name	phone	ssn
박지성	010-5910-2312	900402-1235721

3. 지점 이름이 '성남지점'인 지점을 통해 개설된 모든 예금의 잔액을 검색하라.

```
SELECT SUM(d.balance)
FROM deposit d
WHERE branch_name='성남지점'
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT SUM(d.balance)
2 FROM deposit d
3 WHERE branch_name='성남지점'
4
```

Below the query editor is a 'Result Grid' showing the results of the query:

SUM(d.balance)
▶ 3730000

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

```
SELECT b.branch_head, b.address
FROM branch b
WHERE branch_name='광주지점'
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT b.branch_head, b.address
2 FROM branch b
3 WHERE branch_name='광주지점'
4
5
```

Below the query editor is a 'Result Grid' showing the results of the query:

branch_head	address
▶ 김기백	광주
NULL	NULL

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

```
SELECT branch_name, SUM(balance)
FROM client natural join deposit
WHERE name='김기식'
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT branch_name, SUM(balance)
2 FROM client natural join deposit
3 WHERE name='김기식'
```

Below the query editor is a 'Result Grid' showing the results of the query:

branch_name	SUM(balance)
▶ 서울지점	440000

7. '성남지점'에서 예금이 있는 고객의 이름과 주소, 그리고 예금 잔액을 검색하라.

```
SELECT name, address, SUM(balance) as 예금잔액
FROM client natural join deposit
WHERE deposit.branch_name='성남지점'
GROUP BY ssn
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT name, address, SUM(balance) as 예금잔액
2 FROM client natural join deposit
3 WHERE deposit.branch_name='성남지점'
4 GROUP BY ssn
```

Below the query editor is a 'Result Grid' showing the results of the query. The grid has three columns: name, address, and 예금잔액. The results are as follows:

name	address	예금잔액
이기상	대전	2300000
박지성	서울	870000
박선희	서울	560000

8. '성남지점'에서 예금이 있는 고객 중 박씨 성을 가진 고객의 이름과 예금 잔액을 검색하라.

```
SELECT name, address, SUM(balance) as 예금잔액
FROM client natural join deposit
WHERE branch_name='성남지점' AND name LIKE '박%'
GROUP BY ssn
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT name, address, SUM(balance) as 예금잔액
2 FROM client natural join deposit
3 WHERE branch_name='성남지점' AND name LIKE '박%'
4 GROUP BY ssn
```

Below the query editor is a 'Result Grid' showing the results of the query. The grid has three columns: name, address, and 예금잔액. The results are as follows:

name	address	예금잔액
박지성	서울	870000
박선희	서울	560000

9. 10만원 이상의 예금이 있는 고객의 이름을 검색하라.

```
SELECT name
FROM client
WHERE ssn IN (
    SELECT ssn
    FROM client natural join deposit
    WHERE balance > 100000
    GROUP BY ssn
)
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT name
2 FROM client
3 WHERE ssn IN (
4     SELECT ssn
5     FROM client natural join deposit
6     WHERE balance > 100000
7     GROUP BY ssn
8 )
```

Below the query editor is a 'Result Grid' showing the results of the query. The grid has one column: name. The results are as follows:

name
강지선
홍순태
이기상
박지성
황현희
박선희
김기식
정성태
NULL

10. 10만원 이상의 예금이 있는 고객을 가진 지점의 이름과 지점장 이름을 검색하라.

```
SELECT branch_name, branch_head
FROM branch
WHERE branch_name IN (
    SELECT branch_name
    FROM client natural join deposit
    WHERE balance > 100000
    GROUP BY ssn
)
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT branch_name, branch_head
2 FROM branch
3 WHERE branch_name IN (
4     SELECT branch_name
5     FROM client natural join deposit
6     WHERE balance > 100000
7     GROUP BY ssn
8 )
```

The result grid below the query shows the following data:

branch_name	branch_head
광주지점	김기백
대구지점	김기식
대전지점	이연희
서울지점	강동희
성남지점	박찬주
제주지점	고희경
NULL	NULL

11. 예금을 개설한 지점의 지점장과 이름이 같은 고객이 소유한 예금의 잔액, 개설지점 이름을 검색하라.
(* 동명이인이 있어 이를 구분하였습니다.)

```
SELECT ssn, branch_head, SUM(balance), branch_name
FROM branch natural join deposit
WHERE branch_head IN (
    SELECT branch_head
    FROM branch b, client c
    WHERE c.name = b.branch_head
)
GROUP BY ssn
```

The screenshot shows a SQL query editor with the following query:

```
2 SELECT ssn, branch_head, SUM(balance), branch_name
3 FROM branch natural join deposit
4 WHERE branch_head IN (
5     SELECT branch_head
6     FROM branch b, client c
7     WHERE c.name = b.branch_head
8 )
9 GROUP BY ssn
```

The result grid below the query shows the following data:

ssn	branch_head	SUM(balance)	branch_name
951012-1234123	김기식	9000	대구지점
970102-1234123	김기식	110000	대구지점

13. 주민등록번호 상의 생일이 3월인 모든 고객의 이름과 소유한 예금의 잔액을 검색하라.

```
SELECT name, SUM(balance)
FROM client natural join deposit
WHERE ssn IN (
    SELECT c.ssn
    FROM client c
    WHERE c.ssn LIKE '__03%'
)
GROUP BY ssn
```

The screenshot shows a SQL query editor with the following query:

```
2 SELECT name, SUM(balance)
3 FROM client natural join deposit
4 WHERE ssn IN (
5     SELECT c.ssn
6     FROM client c
7     WHERE c.ssn LIKE '__03%'
8 )
9 GROUP BY ssn
```

The result grid below the query shows the following data:

name	SUM(balance)
홍순태	2080000
정성태	900000

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

```
SELECT name, SUM(balance)
FROM client natural join deposit
WHERE address IN (
    SELECT c.address
    FROM client c, branch b
    WHERE c.address = b.address
)
GROUP BY ssn
```

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

15. '성남지점'과 거래하고 있는 고객의 숫자를 검색하라.

```
SELECT COUNT(d.ssn) as 성남지점고객수
FROM deposit d
WHERE d.branch_name = '성남지점'
```

성남지점고객수
3

16. 각 지점별 잔액의 총합을 검색하라.

```
SELECT d.branch_name, SUM(d.balance) as 예금잔액
FROM deposit d
GROUP BY branch_name
```

branch_name	예금잔액
광주지점	880000
대구지점	119000
대전지점	120000
서울지점	3430000
성남지점	3730000
제주지점	900000

17. 고객 이름별 예금 잔액의 총합을 검색하라.

```
SELECT name, SUM(balance)
FROM client natural join deposit
GROUP BY ssn
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT name, SUM(balance)
2 FROM client natural join deposit
3 GROUP BY ssn
4
5
6
7
```

Below the query editor is the 'Result Grid' showing the results of the query:

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

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

```
SELECT m_d.branch_name, m_balance
FROM (
    SELECT d.ssn, branch_name, SUM(d.balance) as m_balance
    FROM deposit d
    GROUP BY branch_name
) m_d
WHERE m_d.m_balance > 1000000
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT m_d.branch_name, m_balance
2 FROM (
3     SELECT d.ssn, branch_name, SUM(d.balance) as m_balance
4     FROM deposit d
5     GROUP BY branch_name
6 ) m_d
7 WHERE m_d.m_balance > 1000000
8
```

Below the query editor is the 'Result Grid' showing the results of the query:

branch_name	m_balance
▶ 서울지점	3430000
성남지점	3730000

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

```
SELECT *, COUNT(m_d.ssn)
FROM (
    SELECT d.ssn, branch_name, SUM(d.balance) as m_balance
    FROM deposit d
    GROUP BY ssn
) m_d
WHERE m_d.m_balance > 1000000
GROUP BY m_d.branch_name
```

The screenshot shows a SQL query editor with the following query:

```
1 SELECT *, COUNT(m_d.ssn)
2 FROM (
3     SELECT d.ssn, branch_name, SUM(d.balance) as m_balance
4     FROM deposit d
5     GROUP BY ssn
6 ) m_d
7 WHERE m_d.m_balance > 1000000
8 GROUP BY m_d.branch_name
```

Below the query editor is the 'Result Grid' showing the results of the query:

ssn	branch_name	m_balance	COUNT(m_d.ssn)
▶ 890302-1273121	광주지점	2080000	1
910502-2123121	서울지점	1900000	1
890902-1238311	성남지점	2300000	1

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

```
SELECT c.name, c.phone  
FROM client c  
WHERE ssn NOT IN (  
    SELECT ssn  
    FROM client natural join deposit  
    GROUP BY ssn  
)
```

