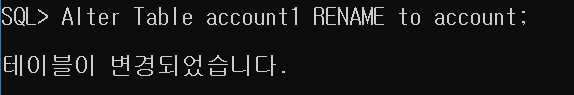
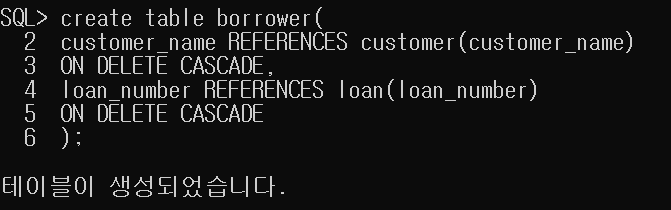
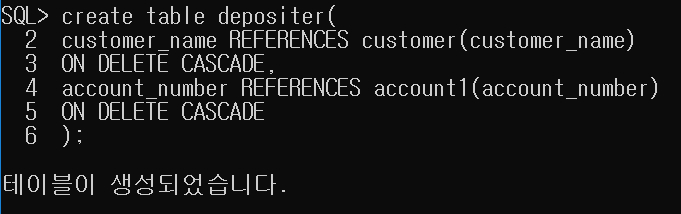
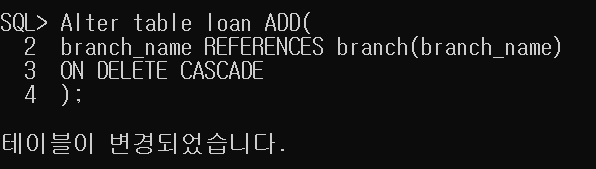
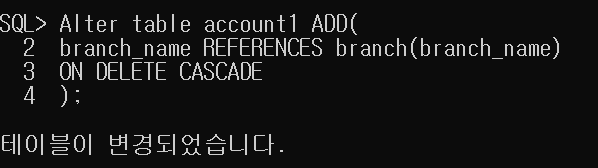
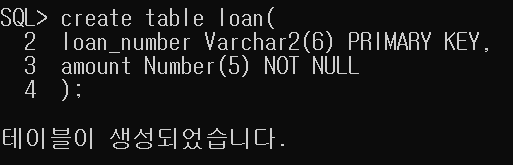
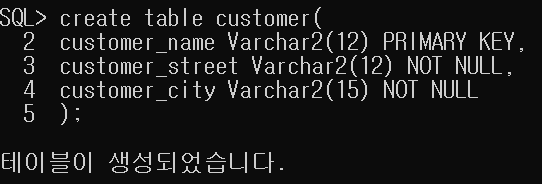
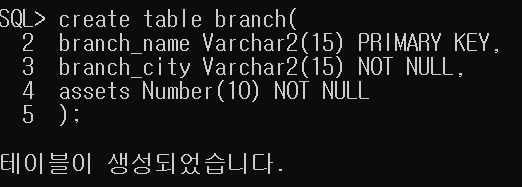
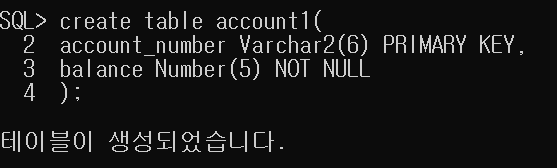
|  |
| --- |
|  |
| 데이터베이스 입문 |
| Project #3 |

|  |
| --- |
| 20141334 임재현 |

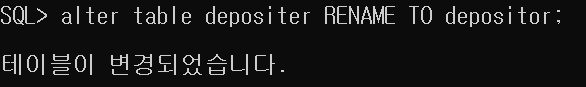
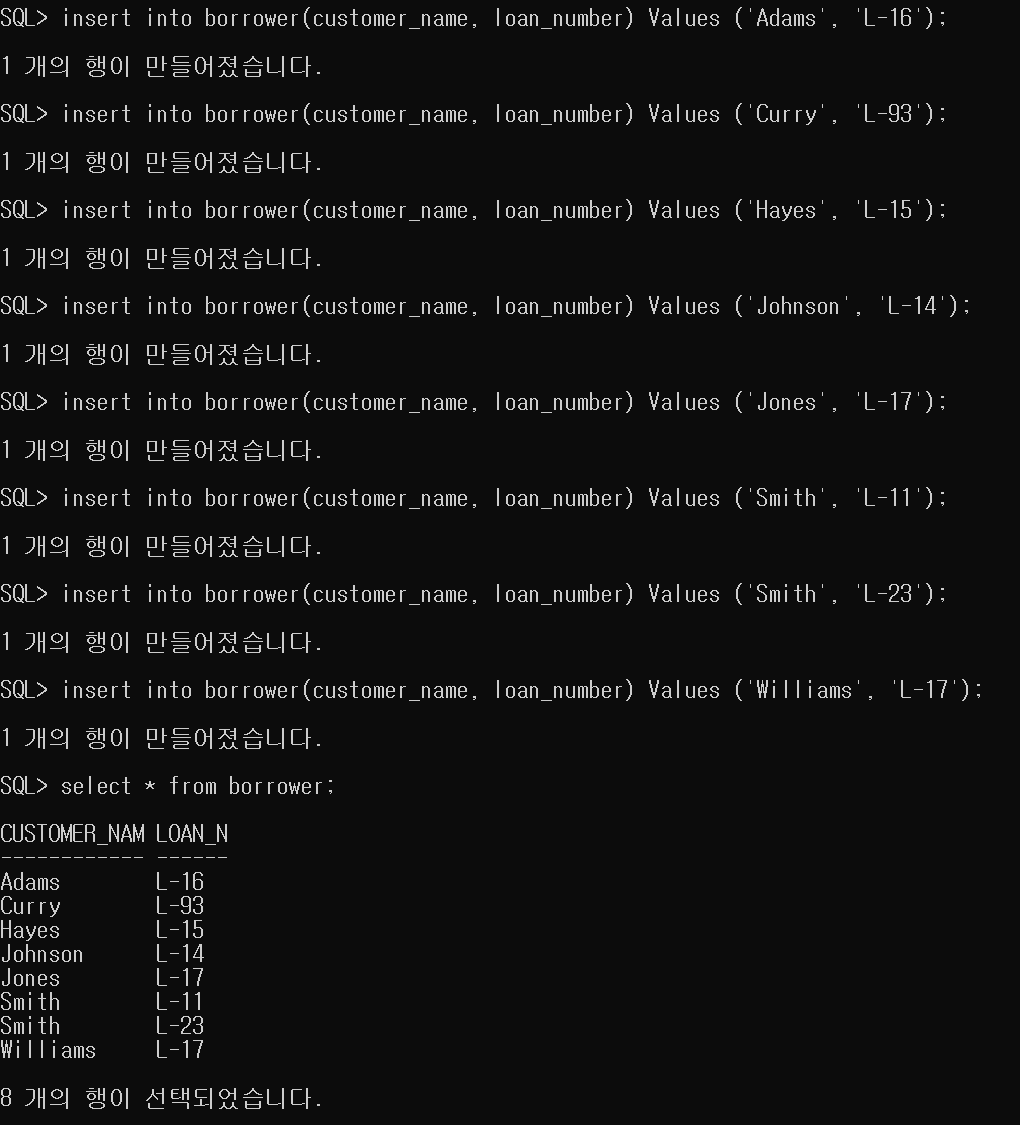
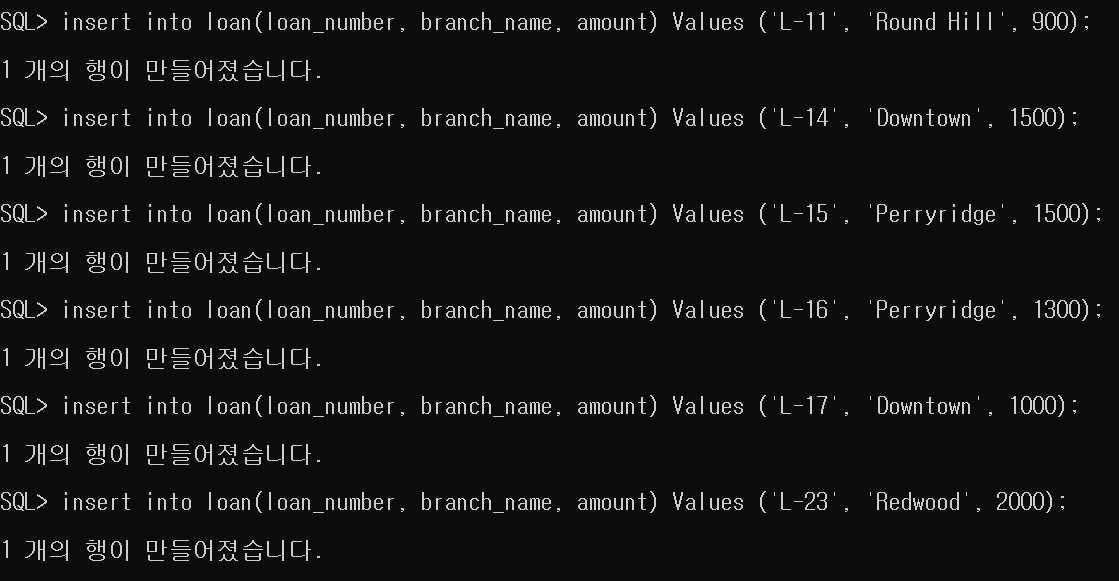
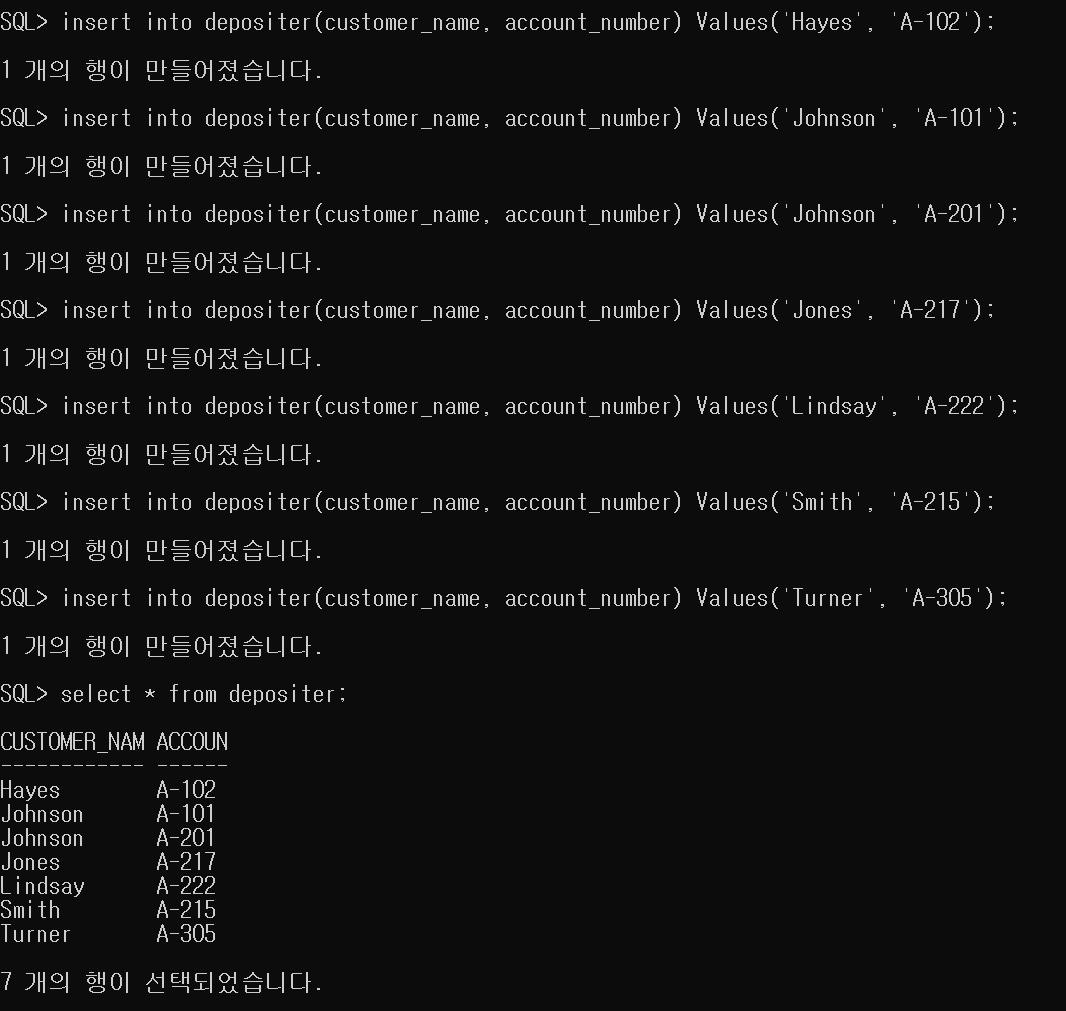
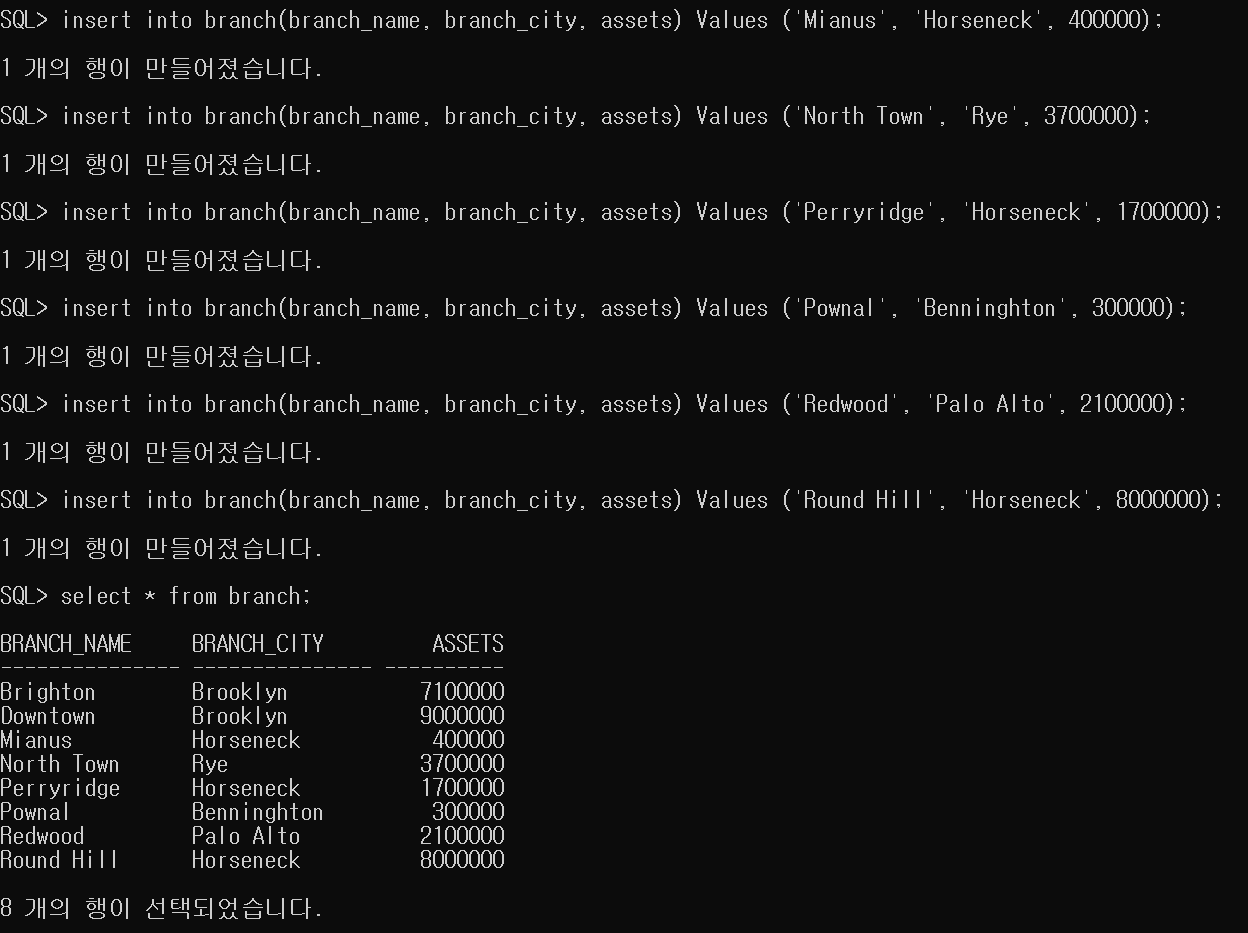
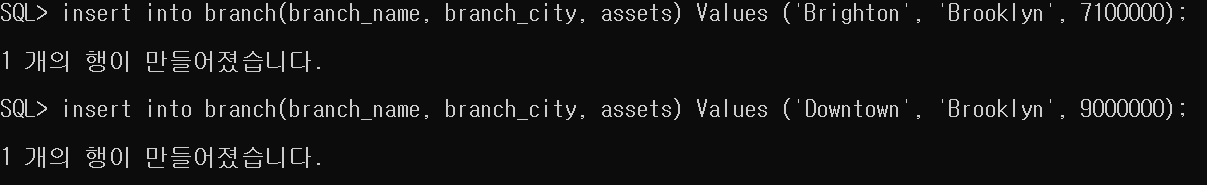
데이터베이스입문 Project #3

20141334 임재현

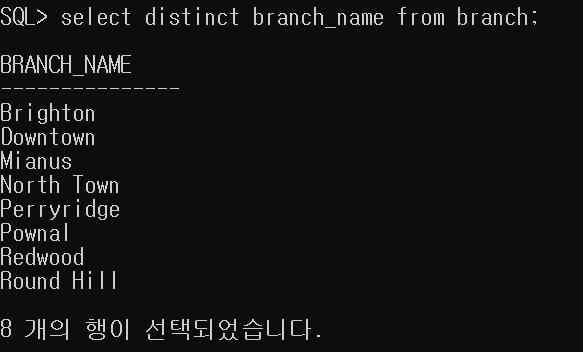
* 아래 번호는 해당 번호의 문제를 수행하기 위한 쿼리문의 내용입니다.

1. E-R 다이어그램을 Oracle DBMS 에 입력하고 테이블 생성결과를 확인하시오.

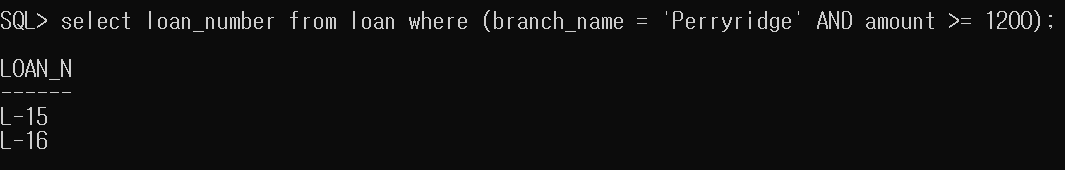
* Primary key를 참조하는 테이블을 후순위로 생성하였고, 작성 당시 과제 #2의 account table과 같은 이름을 가지게 할 수 없어 account1이라는 이름의 테이블로 스키마를 생성후 채점이 끝난 뒤에 해당 스키마를 삭제하고 Rename을 이용하여 테이블 명을 account로 변경하였음.

1. 데이터를 입력하시오 (insert)

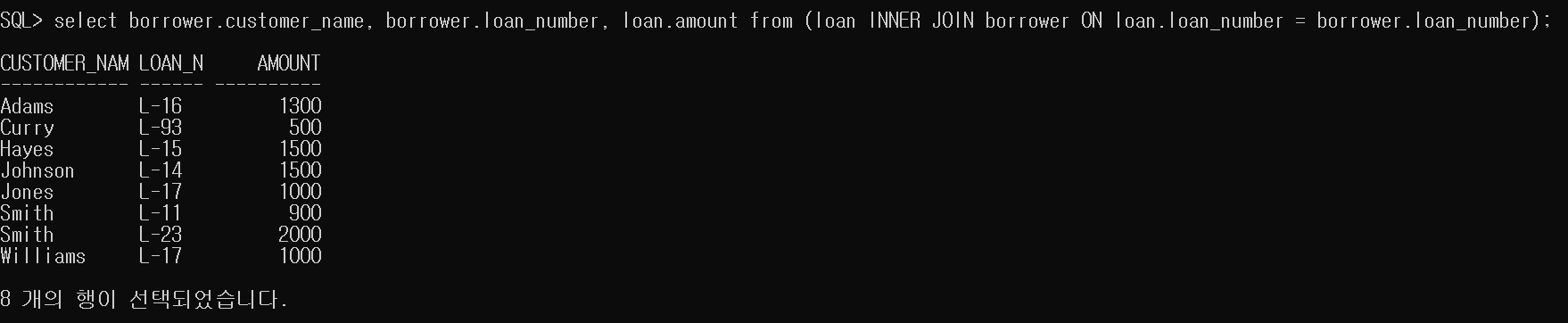
* 테이블명에 오타가 발생해 rename으로 수정하였으며, 각 테이블의 데이터를 알맞게 insert를 이용해 삽입하였다.

1. 중복되지 않은 모든 지점들의 이름을 구하라. (distinct)

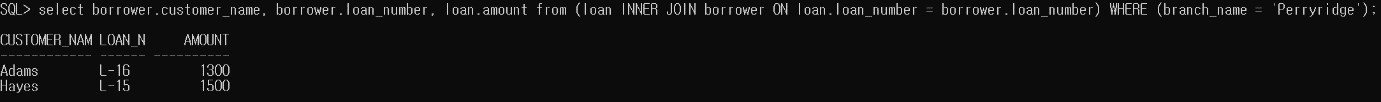
* Distinct 를 이용해 branch\_name을 중복되지 않게 출력하도록 했다.

1. Perryridge 지점에서 $1200 이상의 대출 총액을 지닌 모든 대출에 대해 대출 번호를 전부 구하라. (select)

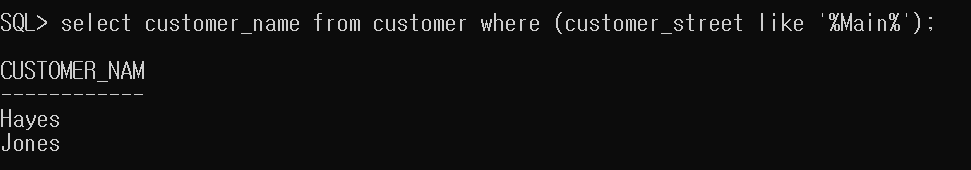
* Select를 이용해 where문의 조건을 만족하는 대출 번호를 출력하도록 했다.

1. 은행에 대출을 가지고 있는 모든 고객들에 대해 그들의 이름과 대출번호와 대출 액수를 구하라. (select)

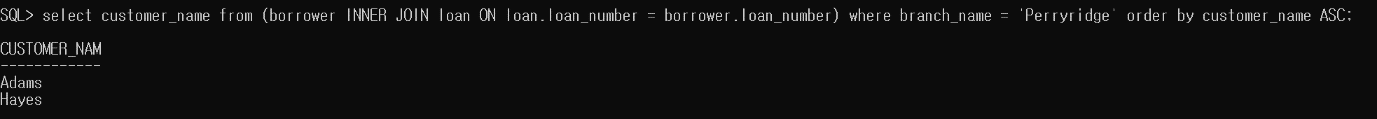
* Select를 이용해 원하는 컬럼의 값들을 불러왔으며, 대출 액수와 이름을 동시에 얻기 위해 inner join을 사용하였다.

1. Perryridge 지점의 모든 대출에 대하여 고객의 이름과 대출 번호, 대출 액수를 구하라.

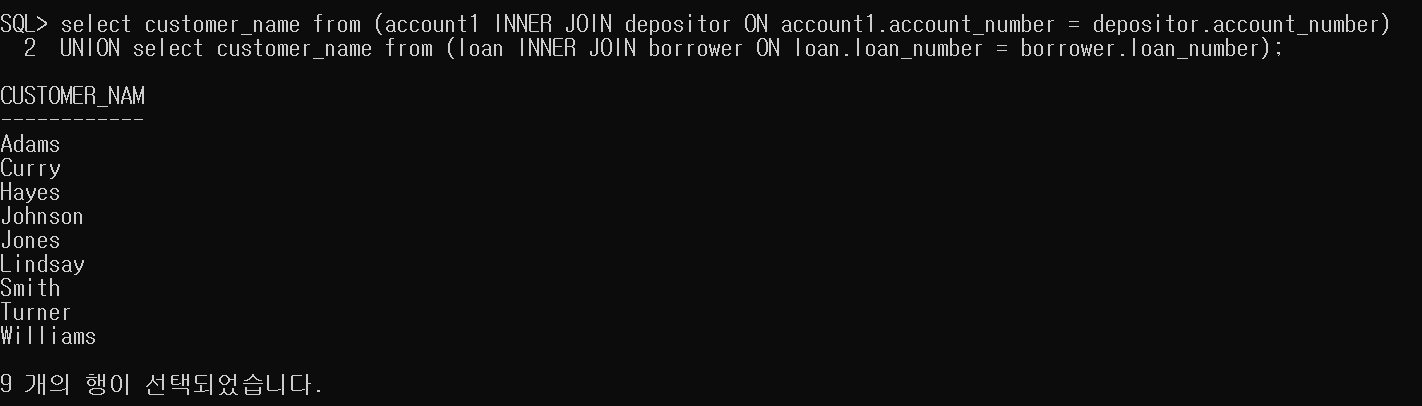
* Inner join을 이용해 고객의 이름과 대출 액수를 매치시켰고, where의 조건에 Perryridge 지점을 넣어 해당 지점의 대출에 대한 값들만 출력하도록 했다.

1. 이름에 'Main'이라는 부분 문자열이 포함된 거리에 살고 있는 모든 고객들의 이름을 구하여라. (like)

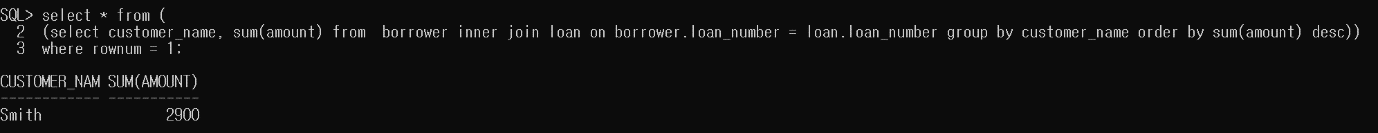
* Like를 이용해 특정 문자열이 포함되어 있다는 조건을 만족하는 인스턴스의 이름만 출력하도록 했다.

1. Perryridge 지점의 대출을 가진 모든 고객들을 알파벳 순서로 나열하라. (order by)

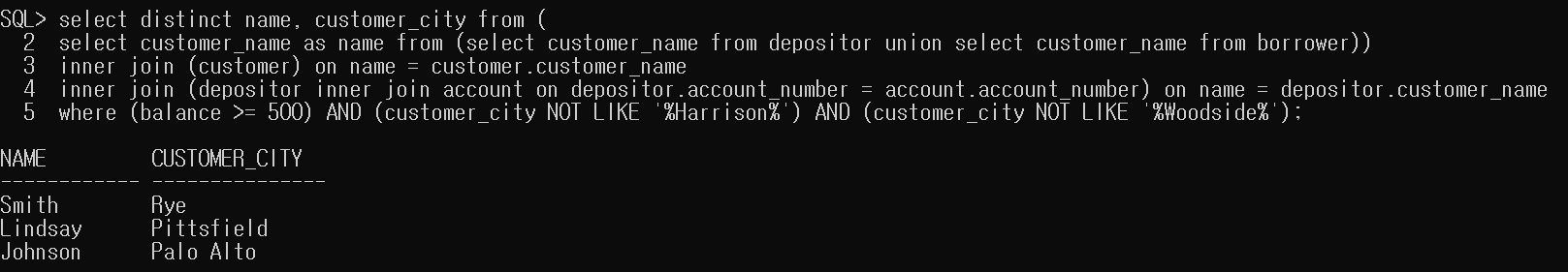
* 조건을 만족하는 고객들에 대해 order by와 asc를 이용해 이름을 알파벳 순으로 정렬하여 출력하도록 했다.

1. 은행에서 대출, 계좌 혹은 둘 다를 가진 모든 고객을 나열하라. (union)

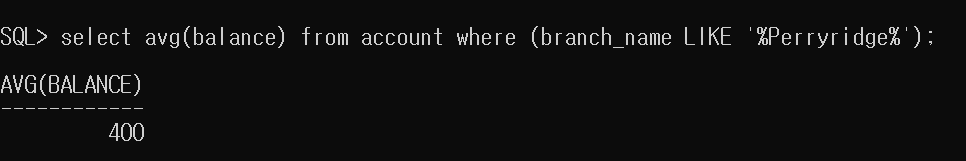
* 대출과 계좌에 대해 각각 Inner join을 이용하여 대출이 있는 고객, 계좌가 있는 고객을 구했고, union을 이용해 위 두개의 고객을 중복없이 합하여 출력하도록 했다.

1. 대출 총액이 가장 큰 고객의 이름과 대출 총액을 구하여라. (max) 

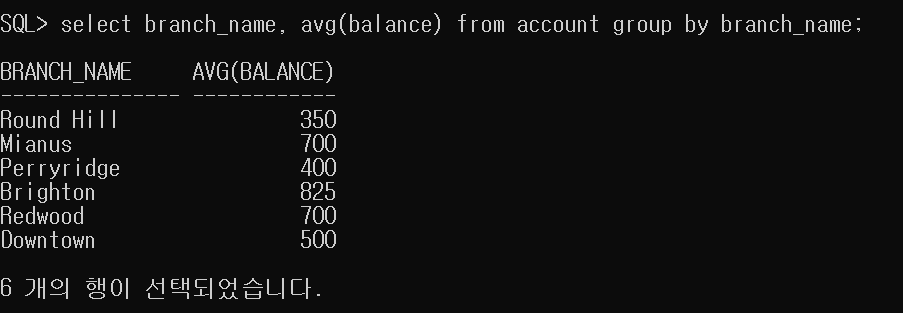
* (Max 키워드 안에 sum 키워드를 중복적용이 불가능하여 다른 방법으로 답만 얻었음) inner join을 이용해 고객의 이름과 대출 총액을 계산, 내림차순으로 정렬해 가장 총액이 큰 한명의 값을 출력하도록 했다.

1. Harrison과 Woodside 에 살지 않으면서 계좌에 잔고가 500이상 있는 고객의 이름과 고객이 사는 도시를 구하라. (select)

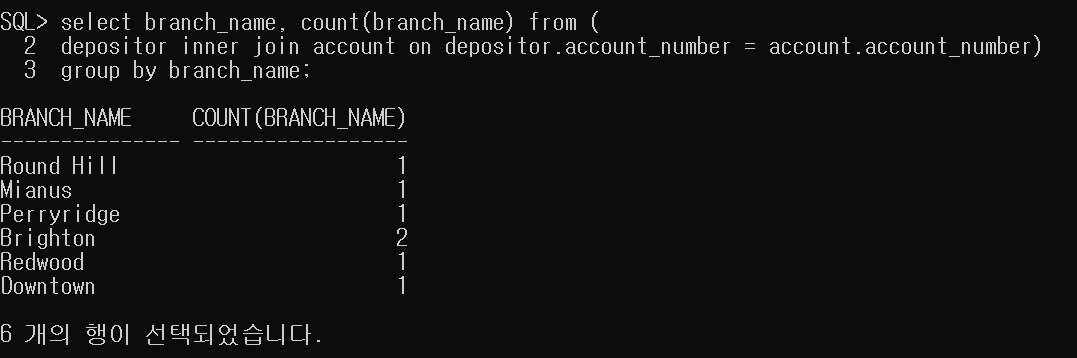
* 고객의 이름과 잔고를 매칭하기 위해 inner join을 이용했고, 그 후 고객의 이름과 사는 도시를 매칭하기 위해 customer와 inner join을 이용했다. 이 후 세 개의 조건에 맞는 값들만 출력하도록 했다.

1. Perryridge 지점에서 계좌의 평균 잔고를 구하여라. (avg)

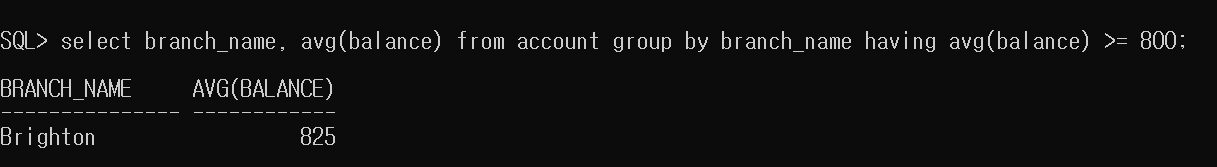
* 조건에 맞는 잔고들의 평균을 avg를 이용해 출력하도록 했다.

1. 각 지점의 평균 계좌 잔고를 구하라. (avg, group by)

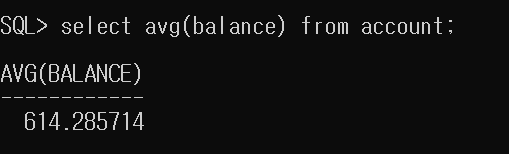
* Avg와 group by를 이용해 branch\_name 마다의 계좌 잔고를 구하도록 했다.

1. 각 지점의 예금자들의 수를 구하라. (count, group by)

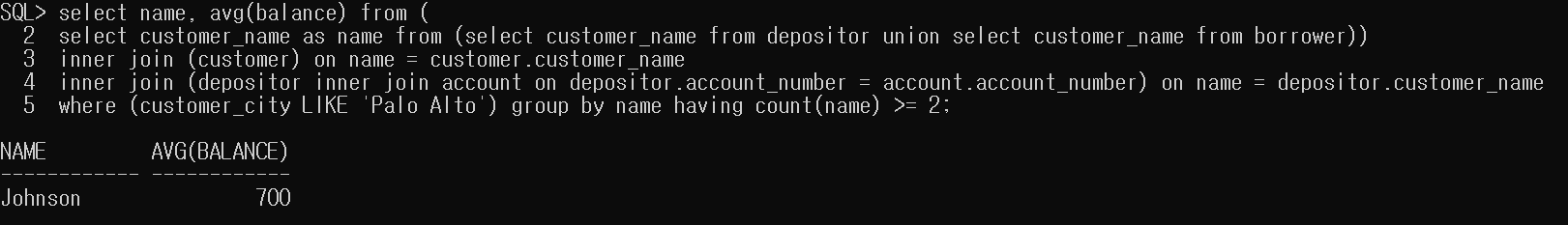
* Inner join을 이용해 예금자와 지점의 이름을 매칭시켰으며, count와 group by를 이용해 지점의 이름이 테이블에 포함된 횟수를 세어 출력하도록 했다.

1. 평균 잔고가 $800 이상인 지점 이름과 평균 잔고를 나열하라. (avg, group by, having)

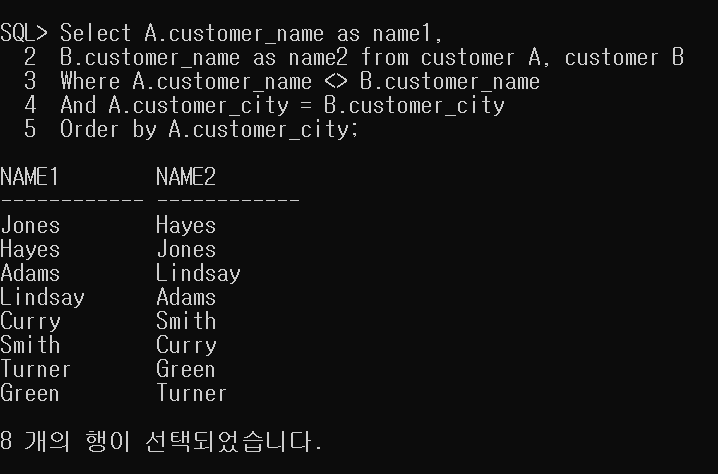
* Avg와 group by를 이용해 지점의 이름당 avg를 계산하도록 하고 where를 이용 할 수 없는 곳에 having을 이용하여 조건을 추가하여 출력하도록 했다.

1. 모든 계좌의 평균 잔고를 구하라. (avg)

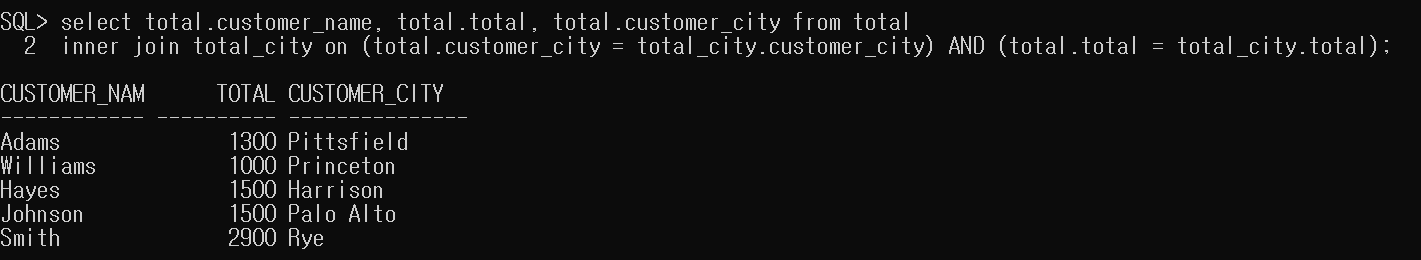
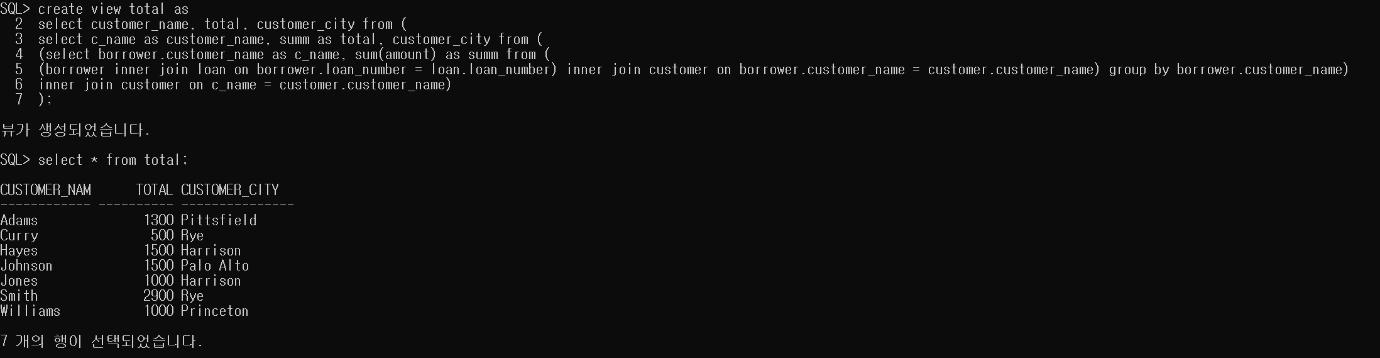
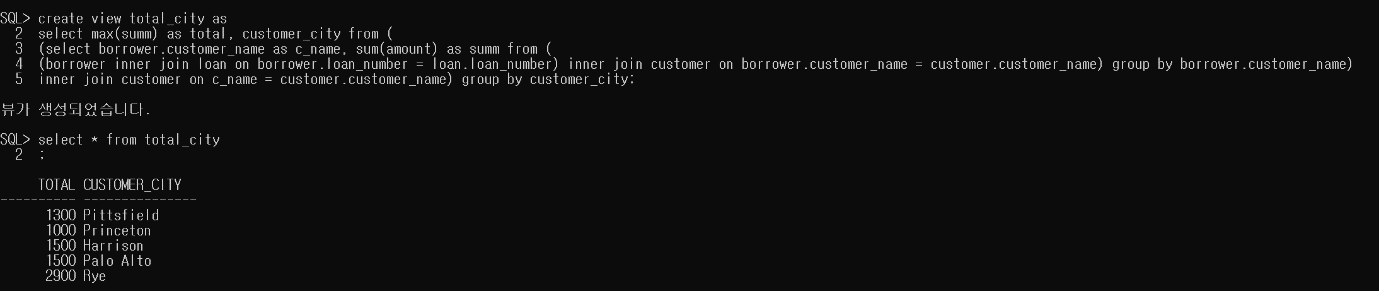
* 계좌에 있는 모든 인스턴스의 잔고의 평균을 avg를 이용해 출력하도록 했다.

1. Palo Alto 에 살고 최소한 두 개의 계좌를 가진 각각의 고객들의 이름과 평균 잔고를 구하라. (group by, having)

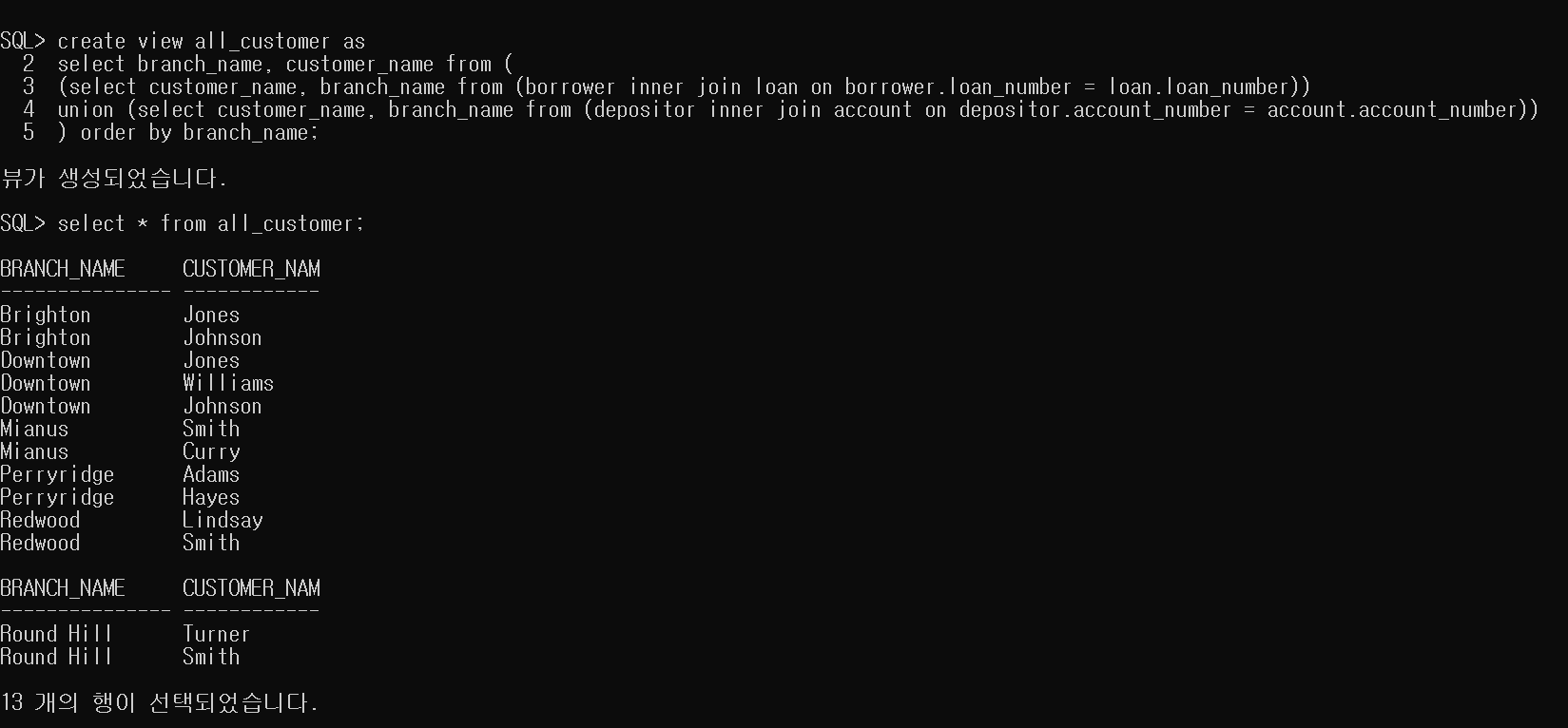
* Inner join을 여러 번 이용하여 고객의 이름과 잔고를 얻어내고, 이후 group by를 이용해 고객당 평균 잔고를 계산하고 having을 이용해 계좌의 개수에 대한 조건을 추가하여 조건을 만족하는 고객의 이름과 평균 잔고를 출력하도록 했다.

1. 같은 도시에 사는 고객의 이름의 쌍을 구하여라. (select)

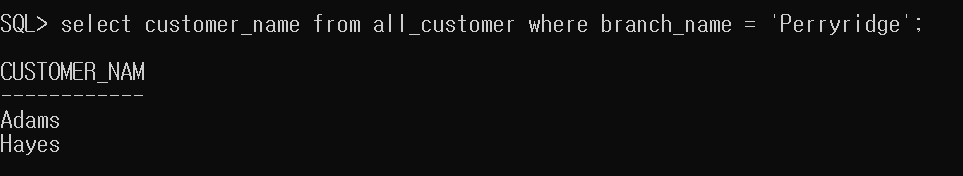
* Self join을 이용해 같은 도시에 사는 고객들의 이름의 순서쌍을 모두 출력하도록 했다.

1. 각 도시 별로 가장 높은 대출 총액을 가지고 있는 고객의 이름과 대출 총액을 구하여라. 단, 대출 을 가진 고객이 살지 않는 도시는 표시하지 않는다. (max, group by)

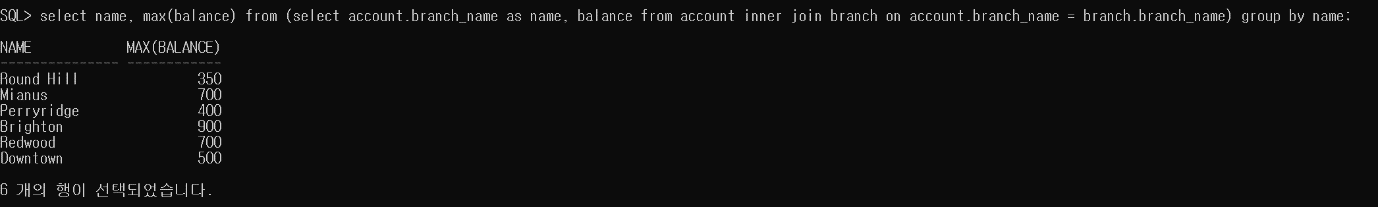
* Max와 group by를 이용해 최대 대출 총액과 도시를 매칭한 view인 total\_city와 고객의 이름, 대출 총액, 도시를 모두 가지고 있는 view인 total을 생성하고, inner join을 이용해 총액과 도시가 일치하는 인스턴스의 값만 출력하도록 했다.

1. 지점 이름과 그 지점에 계좌나 대출 둘 중 하나를 가진 고객 이름으로 구성된 View 를 작성하라. 단 View의 이름은 all\_customer이다. (create view)

* Inner join과 union을 이용한 계좌 및 대출을 가지고 있는 고객을 구하는 쿼리문에 대해 create view를 이용하여 결과값을 새로운 View에 저장하였음.

1. 20에서 생성된 View 를 이용하여 Perryridge 지점의 모든 고객 이름을 나열하라.

* 새로운 뷰를 생성해둠으로써 별도의 복잡한 쿼리문 없이 고객의 이름을 바로 가져와 출력하도록 했다.

1. 각 지점에서 총 잔고의 최대값을 나열하라. (as)

* 중복된 이름의 컬럼이 다수 있어 as를 이용해 alias를 지정하여 이후 select가 수월하도록 했다.