

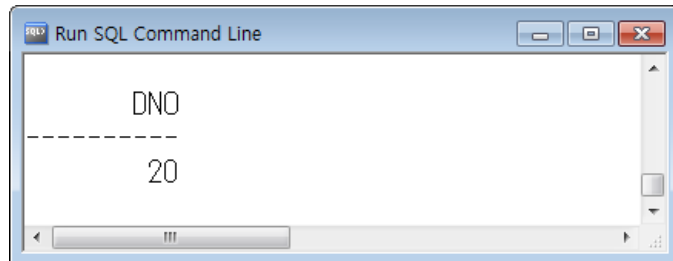
# 학습 내용

- ❖ 조인
- ❖ EQUI JOIN
- ❖ NON-EQUI JOIN
- ❖ SEIF JOIN
- ❖ OUTER JOIN

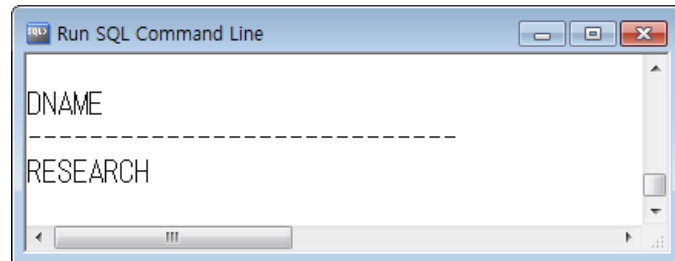
## 01. 조인

- ❖ 여러 테이블에 저장된 데이터를 한 번에 조회해야 할 필요가 있을 때 사용
- ❖ 7788인 사원의 이름과 소속 부서명을 출력하려고 한다

```
select dno  
예 from employee  
where eno=7788;
```



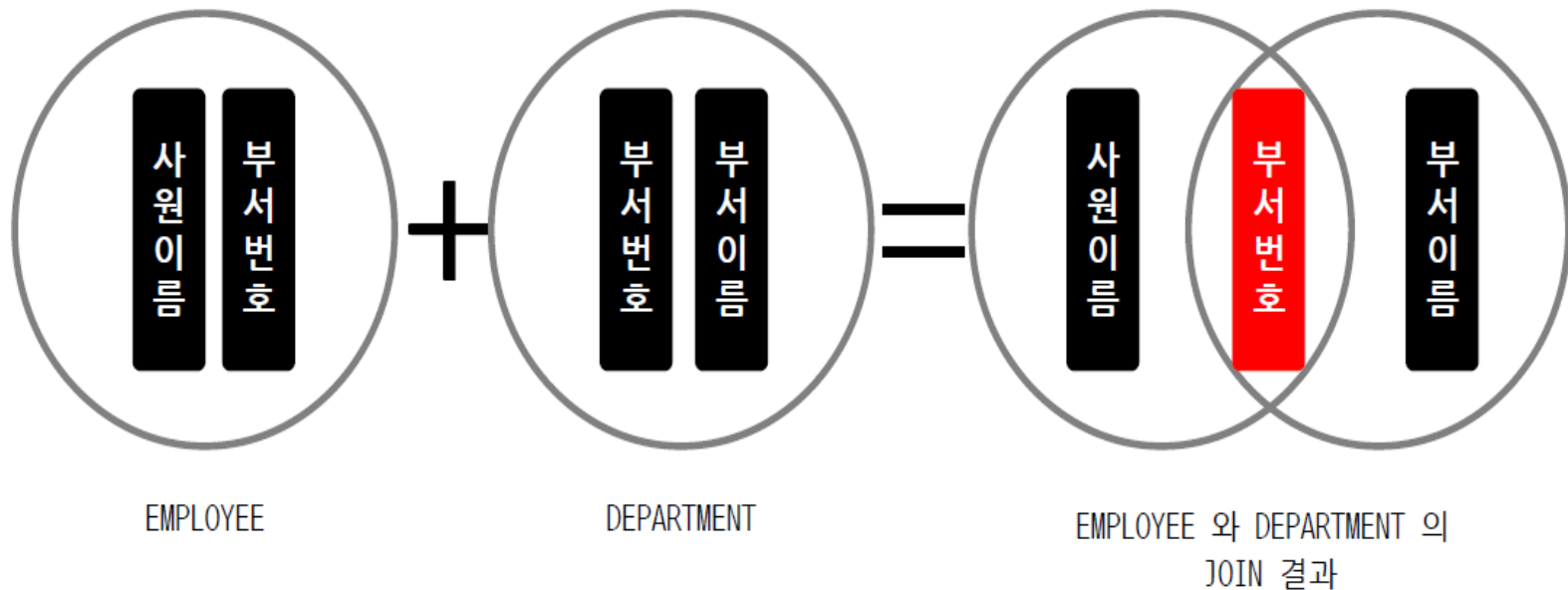
```
select dname  
예 from department  
where dno=20;
```



# 01. 조인

서로 연관되고 다른 테이블에 존재하는 컬럼들을 한번에 조회하기 위해 사용하는 대표적인 기법

[JOIN 개념 도식 1]



# 카디시안 곱

- ❖ 특별한 키워드 없이 SELECT 문의 FROM 절에 사원(employee) 테이블과 부서(department) 테이블을 콤마로 연결하여 연속하여 기술

Run SQL Command Line

```
SQL> select * from department, employee;
```

DNO	DNAME	LOC	ENO	ENAME	JOB	MANAGER	HIREDATE	SALARY	COMMISSION	DNO
10	ACCOUNTING	NEW YORK	7369	SMITH	CLERK	7902	80/12/17	800		20
10	ACCOUNTING	NEW YORK	7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
10	ACCOUNTING	NEW YORK	7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
10	ACCOUNTING	NEW YORK	7566	JONES	MANAGER	7839	81/04/02	2975		20
10	ACCOUNTING	NEW YORK	7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
10	ACCOUNTING	NEW YORK	7698	BLAKE	MANAGER	7839	81/05/01	2850		30
10	ACCOUNTING	NEW YORK	7782	CLARK	MANAGER	7839	81/06/09	2450		10
10	ACCOUNTING	NEW YORK	7788	SCOTT	ANALYST	7566	87/07/13	3000		20
10	ACCOUNTING	NEW YORK	7839	KING	PRESIDENT		81/11/17	5000		10
10	ACCOUNTING	NEW YORK	7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
10	ACCOUNTING	NEW YORK	7876	ADAMS	CLERK	7788	87/07/13	1100		20
10	ACCOUNTING	NEW YORK	7900	JAMES	CLERK	7698	81/12/03	950		30
10	ACCOUNTING	NEW YORK	7902	FORD	ANALYST	7566	81/12/03	3000		20
10	ACCOUNTING	NEW YORK	7934	MILLER	CLERK	7782	82/01/23	1300		10
20	RESEARCH	DALLAS	7369	SMITH	CLERK	7902	80/12/17	800		20
20	RESEARCH	DALLAS	7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
20	RESEARCH	DALLAS	7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
20	RESEARCH	DALLAS	7566	JONES	MANAGER	7839	81/04/02	2975		20
20	RESEARCH	DALLAS	7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
20	RESEARCH	DALLAS	7698	BLAKE	MANAGER	7839	81/05/01	2850		30
20	RESEARCH	DALLAS	7782	CLARK	MANAGER	7839	81/06/09	2450		10
20	RESEARCH	DALLAS	7788	SCOTT	ANALYST	7566	87/07/13	3000		20
20	RESEARCH	DALLAS	7839	KING	PRESIDENT		81/11/17	5000		10
20	RESEARCH	DALLAS	7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
20	RESEARCH	DALLAS	7876	ADAMS	CLERK	7788	87/07/13	1100		20
20	RESEARCH	DALLAS	7900	JAMES	CLERK	7698	81/12/03	950		30
20	RESEARCH	DALLAS	7902	FORD	ANALYST	7566	81/12/03	3000		20
20	RESEARCH	DALLAS	7934	MILLER	CLERK	7782	82/01/23	1300		10
30	SALES	CHICAGO	7369	SMITH	CLERK	7902	80/12/17	800		20
30	SALES	CHICAGO	7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
30	SALES	CHICAGO	7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
30	SALES	CHICAGO	7566	JONES	MANAGER	7839	81/04/02	2975		20
30	SALES	CHICAGO	7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
30	SALES	CHICAGO	7698	BLAKE	MANAGER	7839	81/05/01	2850		30
30	SALES	CHICAGO	7782	CLARK	MANAGER	7839	81/06/09	2450		10
30	SALES	CHICAGO	7788	SCOTT	ANALYST	7566	87/07/13	3000		20
30	SALES	CHICAGO	7839	KING	PRESIDENT		81/11/17	5000		10
30	SALES	CHICAGO	7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
30	SALES	CHICAGO	7876	ADAMS	CLERK	7788	87/07/13	1100		20
30	SALES	CHICAGO	7900	JAMES	CLERK	7698	81/12/03	950		30
30	SALES	CHICAGO	7902	FORD	ANALYST	7566	81/12/03	3000		20
30	SALES	CHICAGO	7934	MILLER	CLERK	7782	82/01/23	1300		10
40	OPERATIONS	BOSTON	7369	SMITH	CLERK	7902	80/12/17	800		20
40	OPERATIONS	BOSTON	7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
40	OPERATIONS	BOSTON	7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
40	OPERATIONS	BOSTON	7566	JONES	MANAGER	7839	81/04/02	2975		20
40	OPERATIONS	BOSTON	7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
40	OPERATIONS	BOSTON	7698	BLAKE	MANAGER	7839	81/05/01	2850		30
40	OPERATIONS	BOSTON	7782	CLARK	MANAGER	7839	81/06/09	2450		10
40	OPERATIONS	BOSTON	7788	SCOTT	ANALYST	7566	87/07/13	3000		20
40	OPERATIONS	BOSTON	7839	KING	PRESIDENT		81/11/17	5000		10
40	OPERATIONS	BOSTON	7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
40	OPERATIONS	BOSTON	7876	ADAMS	CLERK	7788	87/07/13	1100		20
40	OPERATIONS	BOSTON	7900	JAMES	CLERK	7698	81/12/03	950		30
40	OPERATIONS	BOSTON	7902	FORD	ANALYST	7566	81/12/03	3000		20
40	OPERATIONS	BOSTON	7934	MILLER	CLERK	7782	82/01/23	1300		10

56 rows selected.

# Equi Join

- ❖ 조인 대상 테이블에서 공통 칼럼을 '=' (equal) 비교를 통해 같은 값을 가지는 행을 연결

형식	<pre>SELECT table1.column, table2.column FROM table1, table2 WHERE table1.column1 = table2.column2;</pre>
----	---

# Equi Join

## ❖ 각 직원들이 소속된 부서정보 얻기

예

```
select *  
from employee, department  
where employee.dno = department.dno;
```

ENO	ENAME	JOB	MANAGER	HIREDATE	SALARY	COMMISSION	DNO	DNO	DNAME	LOC
7782	CLARK	MANAGER	7839	81/06/09	2450		10	10	ACCOUNTING	NEW YORK
7839	KING	PRESIDENT		81/11/17	5000		10	10	ACCOUNTING	NEW YORK
7934	MILLER	CLERK	7782	82/01/23	1300		10	10	ACCOUNTING	NEW YORK
7566	JONES	MANAGER	7839	81/04/02	2975		20	20	RESEARCH	DALLAS
7902	FORD	ANALYST	7566	81/12/03	3000		20	20	RESEARCH	DALLAS
7876	ADAMS	CLERK	7788	87/07/13	1100		20	20	RESEARCH	DALLAS
7369	SMITH	CLERK	7902	80/12/17	800		20	20	RESEARCH	DALLAS
7788	SCOTT	ANALYST	7566	87/07/13	3000		20	20	RESEARCH	DALLAS
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30	30	SALES	CHICAGO
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30	30	SALES	CHICAGO
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30	30	SALES	CHICAGO
7900	JAMES	CLERK	7698	81/12/03	950		30	30	SALES	CHICAGO
7698	BLAKE	MANAGER	7839	81/05/01	2850		30	30	SALES	CHICAGO
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30	30	SALES	CHICAGO

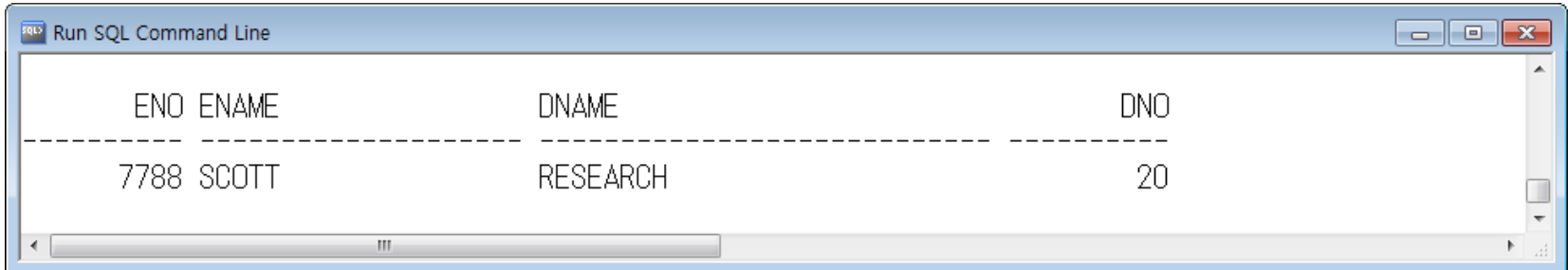
14 rows selected.

## 모호한 칼럼명 자세히 지정

- ❖ WHERE 절에 모호성을 방지하기 위해서 칼럼명 앞에 테이블 이름을 명시

예

```
select employee.eno, employee.ename,  
       department.dname, employee.dno  
from employee, department  
where employee.dno = department.dno  
and employee.eno=7788;
```



The screenshot shows a window titled "Run SQL Command Line" with a table of results. The table has four columns: ENO, ENAME, DNAME, and DNO. The first row shows the data for employee 7788, SCOTT, in the RESEARCH department (DNO 20).

ENO	ENAME	DNAME	DNO
7788	SCOTT	RESEARCH	20

## 테이블에 별칭 사용하기

### ❖ 테이블명이 너무 긴 경우에 테이블명을 대신하는 별칭을 사용

- . 테이블의 별명은 30자까지 가능하지만 너무 길지 않게 작성합니다.
- . FROM 절에서 테이블명을 명시하고 공백을 둔 다음 테이블 별칭을 지정합니다.
- . 하나의 SQL 명령문에서 테이블명과 별명을 혼용할 수 없습니다.
- . 테이블의 별칭은 해당 SQL 명령문 내에서만 유효합니다.

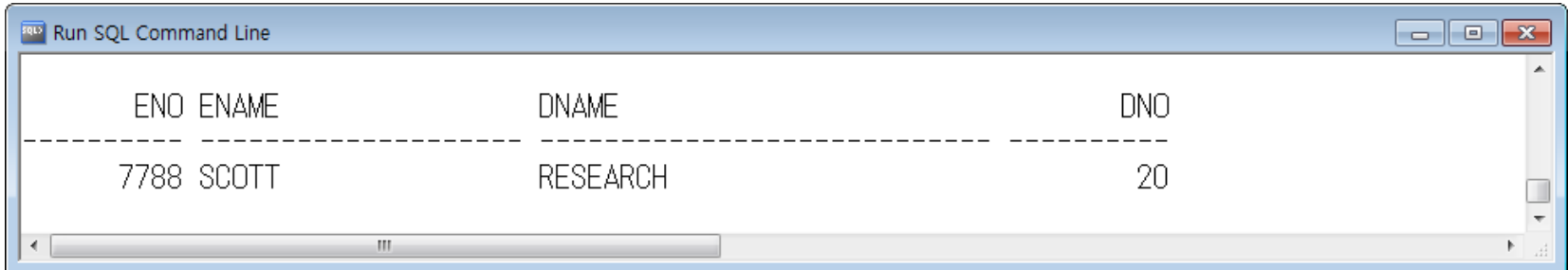


# 테이블에 별칭 사용하기

❖ FROM 절에서 테이블명 다음에 공백을 두고 별명을 정의

예

```
select e.eno, e.ename, d.dname, e.dno  
from employee e, department d  
where e.dno = d.dno  
and e.eno=7788;
```



The screenshot shows a window titled "Run SQL Command Line". Inside, the query results are displayed in a table format with four columns: ENO, ENAME, DNAME, and DNO. The first row shows the data for employee 7788, SCOTT, who works in the RESEARCH department (DNO 20).

ENO	ENAME	DNAME	DNO
7788	SCOTT	RESEARCH	20

# EQUI JOIN–NATURAL JOIN

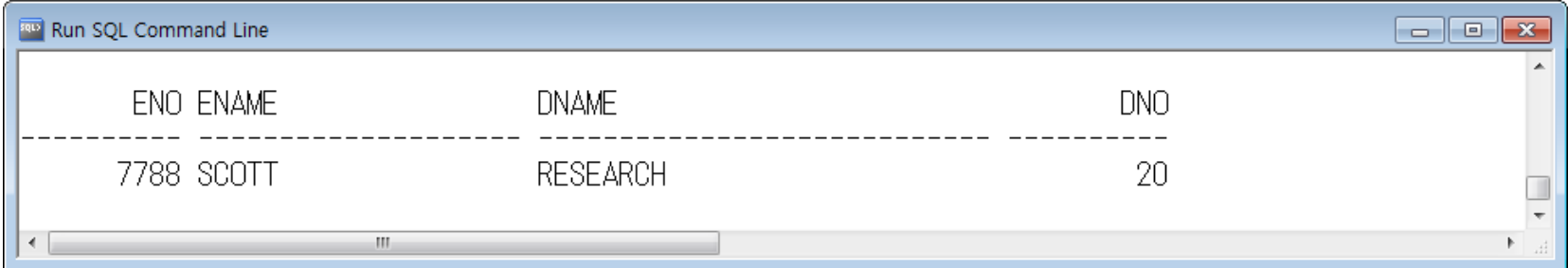
- ❖ **NATURAL JOIN** 키워드를 사용하면 오라클에서 자동적으로 테이블의 모든 칼럼을 대상으로 일치하는 데이터 유형 및 이름을 가진 공통 칼럼을 조사한 후에 자동으로 조인을 수행

형식

```
SELECT table1.column, table2.column  
FROM table1 NATURAL JOIN table2;
```

예

```
select e.eno, e.ename, d.dname, dno  
from employee e natural join department d  
where e.eno=7788;
```



ENO	ENAME	DNAME	DNO
7788	SCOTT	RESEARCH	20

# EQUI JOIN-JOIN ~ USING

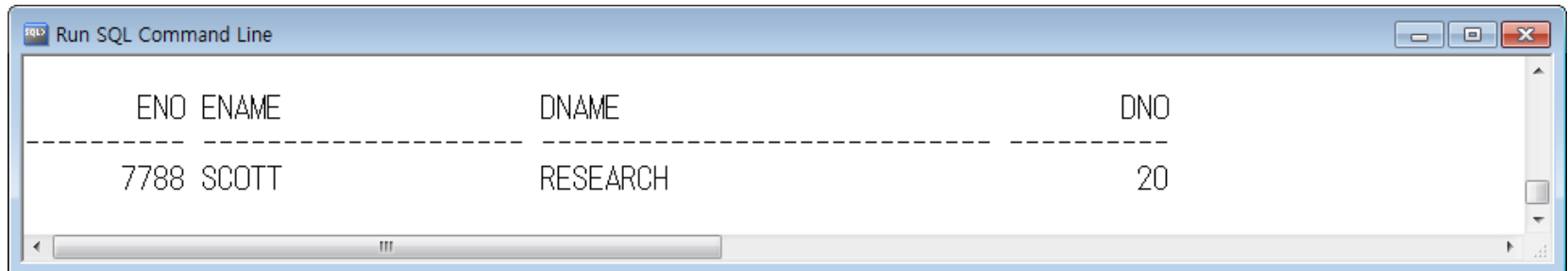
## ❖ USING 절에 조인 대상이 되는 칼럼을 지정

형식

```
SELECT table1.column, table2.column  
FROM table1 JOIN table2  
USING(column);
```

예

```
select e.eno, e.ename, d.dname, dno  
from employee e join department d  
using(dno)  
where e.eno=7788;
```



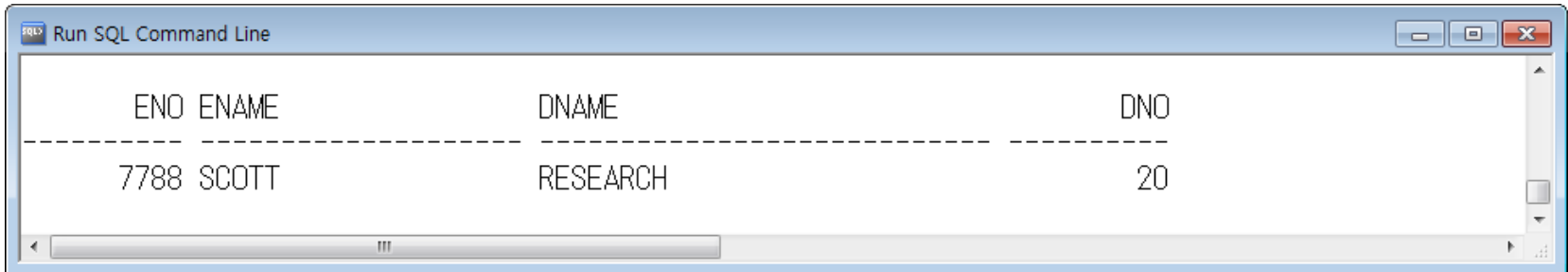
ENO	ENAME	DNAME	DNO
7788	SCOTT	RESEARCH	20

## EQUI JOIN-JOIN ~ ON

- ❖ 임의의 조건을 지정하거나 조인할 칼럼을 지정하려면 ON 절을 사용

예

```
select e.eno, e.ename, d.dname, e.dno  
from employee e join department d  
on e.dno = d.dno  
where e.eno=7788;
```



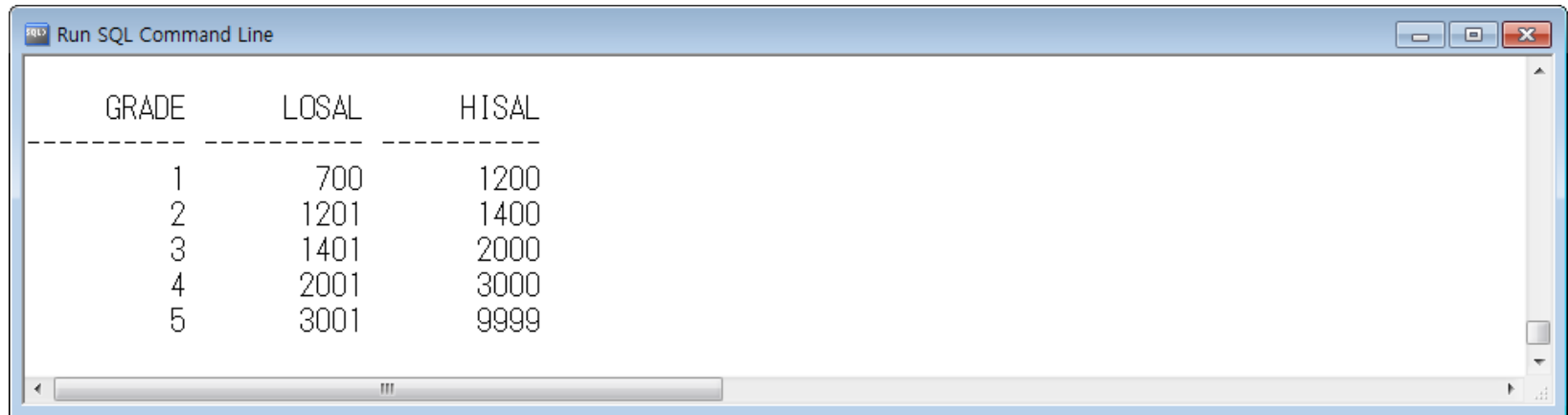
The screenshot shows a window titled "Run SQL Command Line" with a table of query results. The table has four columns: ENO, ENAME, DNAME, and DNO. The data row shows ENO as 7788, ENAME as SCOTT, DNAME as RESEARCH, and DNO as 20.

ENO	ENAME	DNAME	DNO
7788	SCOTT	RESEARCH	20

## 03. Non-Equi Join

- ❖ WHERE 절에 ‘<’ , BETWEEN a AND b와 같이 ‘=’ 조건이 아닌 연산자를 사용
- ❖ 급여 등급 테이블(SALGRADE )을 살펴기

예 **SELECT \* FROM SALGRADE;**



The screenshot shows a window titled "Run SQL Command Line" with a table of salary grades. The table has three columns: GRADE, LOSAL, and HISAL. The data is as follows:

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

## 04. Non-Equi Join

예

```
select ename, salary, grade
from employee, salgrade
where salary between losal and hisal;
```

ENAME	SALARY	GRADE
SMITH	800	1
JAMES	950	1
ADAMS	1100	1
WARD	1250	2
MARTIN	1250	2
MILLER	1300	2
TURNER	1500	3
ALLEN	1600	3
CLARK	2450	4
BLAKE	2850	4
JONES	2975	4
SCOTT	3000	4
FORD	3000	4
KING	5000	5

14 rows selected.

## 3개의 테이블을 조인하기

예

```
select e.ename, d.dname, e.salary, s.grade
from employee e, department d, salgrade s
where e.dno = d.dno
and salary between losal and hisal;
```

ENAME	DNAME	SALARY	GRADE
KING	ACCOUNTING	5000	5
FORD	RESEARCH	3000	4
SCOTT	RESEARCH	3000	4
JONES	RESEARCH	2975	4
BLAKE	SALES	2850	4
CLARK	ACCOUNTING	2450	4
ALLEN	SALES	1600	3
TURNER	SALES	1500	3
MILLER	ACCOUNTING	1300	2
WARD	SALES	1250	2
MARTIN	SALES	1250	2
ADAMS	RESEARCH	1100	1
JAMES	SALES	950	1
SMITH	RESEARCH	800	1

14 rows selected.

## 04. SELF JOIN

- ❖ 하나의 테이블에 있는 칼럼끼리 연결해야 하는 조인이 필요한 경우 사용

예

```
select employees.ename as "사원이름",  
       manager.ename as "직속상관이름"  
from employee employees, employee manager  
where employees.manager = manager.eno;
```

사원이름	직속상관이름
FORD	JONES
SCOTT	JONES
TURNER	BLAKE
ALLEN	BLAKE
WARD	BLAKE
JAMES	BLAKE
MARTIN	BLAKE
MILLER	CLARK
ADAMS	SCOTT
BLAKE	KING
JONES	KING
CLARK	KING
SMITH	FORD

13 rows selected.



## 05. OUTER JOIN

- ❖ EQUI JOIN에서 양측 칼럼 값 중의 하나가 NULL이지만 조인 결과로 출력할 필요가 있는 경우에 OUTER JOIN을 사용

형식

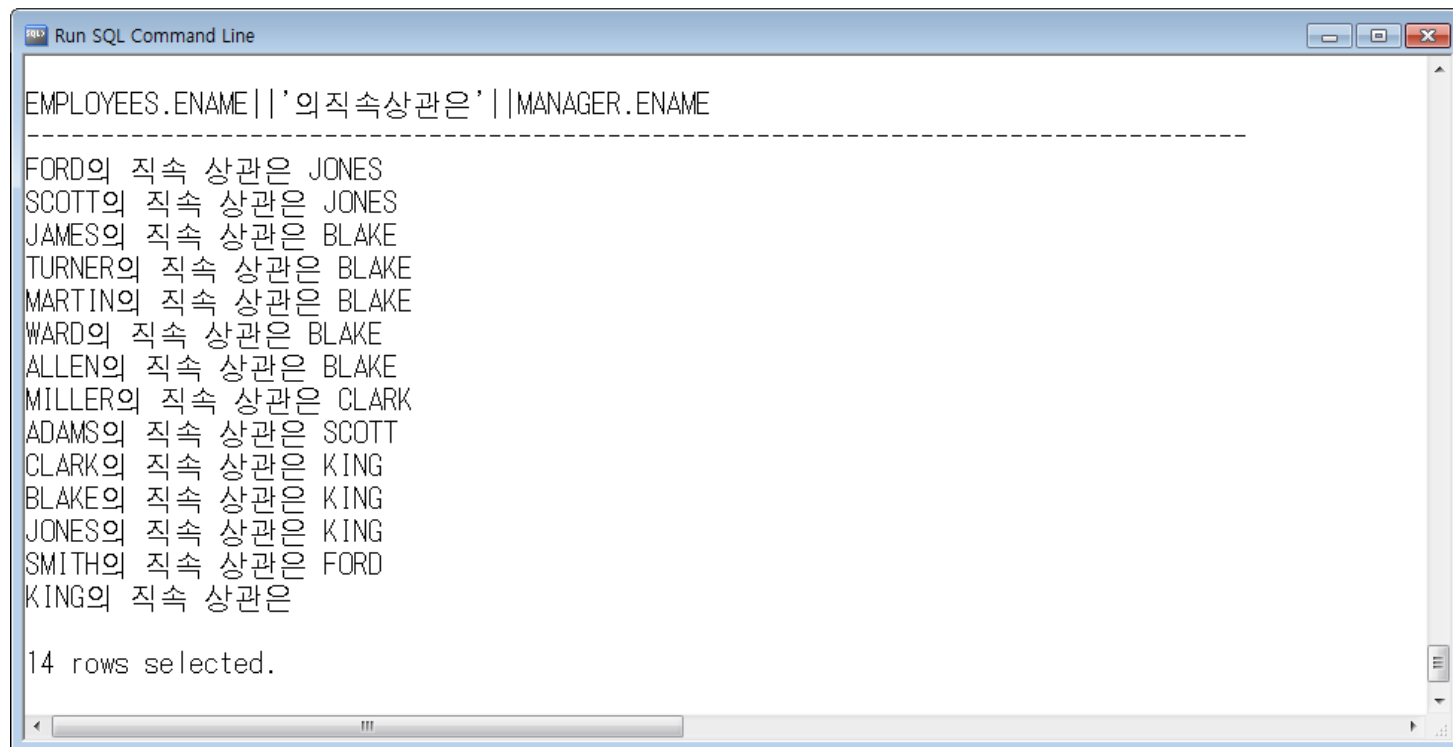
```
SELECT table1.column, table2.column  
FROM table1, table2  
WHERE table1.column(+) = table2.column;  
또는  
table1.column = table2.column(+)
```

## 05. OUTER JOIN

### ❖ (+) 기호를 사용한 OUTER JOIN

예

```
select employees.ename || '의 직속 상관은 ' || manager.e  
name  
from employee employees join employee manager  
on employees.manager = manager.eno(+);
```



```
Run SQL Command Line
```

```
EMPLOYEES.ENAME || '의 직속상관은' || MANAGER.ENAME
```

```
-----
```

```
FORD의 직속 상관은 JONES
```

```
SCOTT의 직속 상관은 JONES
```

```
JAMES의 직속 상관은 BLAKE
```

```
TURNER의 직속 상관은 BLAKE
```

```
MARTIN의 직속 상관은 BLAKE
```

```
WARD의 직속 상관은 BLAKE
```

```
ALLEN의 직속 상관은 BLAKE
```

```
MILLER의 직속 상관은 CLARK
```

```
ADAMS의 직속 상관은 SCOTT
```

```
CLARK의 직속 상관은 KING
```

```
BLAKE의 직속 상관은 KING
```

```
JONES의 직속 상관은 KING
```

```
SMITH의 직속 상관은 FORD
```

```
KING의 직속 상관은
```

```
14 rows selected.
```

# ANSI Outer Join

- ❖ ANSI 구문에서 Outer Join은 LEFT Outer Join, RIGHT Outer Join 그리고 FULL Outer Join 세 가지 타입의 조인을 제공합니다.

**형식**

```
SELECT table1.column, table2.column  
FROM table1[RIGHT | LEFT | FULL] OUTER JOIN table2  
ON table1.column = table2.column;
```

# ANSI Outer Join

- ❖ LEFT OUTER JOIN은 FROM 절의 왼쪽에 위치한 테이블이 NULL을 가질 경우에 사용합니다.

**예** `select employees.ename || '의 직속 상관은 ' || manager.ename  
from employee employees left outer join employee manager  
on employees.manager = manager.eno;`

```
Run SQL Command Line

SQL> select employees.ename || '의 직속 상관은 ' || manager.ename
2  from employee employees left outer join employee manager
3  on employees.manager = manager.eno;

EMPLOYEES.ENAME||'의 직속상관은'||MANAGER.ENAME
-----
FORD의 직속 상관은 JONES
SCOTT의 직속 상관은 JONES
JAMES의 직속 상관은 BLAKE
TURNER의 직속 상관은 BLAKE
MARTIN의 직속 상관은 BLAKE
WARD의 직속 상관은 BLAKE
ALLEN의 직속 상관은 BLAKE
MILLER의 직속 상관은 CLARK
ADAMS의 직속 상관은 SCOTT
CLARK의 직속 상관은 KING
BLAKE의 직속 상관은 KING

EMPLOYEES.ENAME||'의 직속상관은'||MANAGER.ENAME
-----
JONES의 직속 상관은 KING
SMITH의 직속 상관은 FORD
KING의 직속 상관은

14 rows selected.
```

## JOIN - ANSI 표준 구문

- JOIN 유형을 세분화
- WHERE 절에서 JOIN 조건을 별도로 분리하고 'JOIN' 키워드를 명시적으로 사용

```
SELECT ...  
FROM   table1  
{[INNER] JOIN table2 ON (condition1 [AND condition2 ...]) |  
 [INNER] JOIN table2 USING (column1 [, ...]) |  
 NATURAL [INNER] JOIN table2 |  
 LEFT|RIGHT|FULL [OUTER] JOIN table2 ON (condition1 [AND condition2 ...]) |  
 LEFT|RIGHT|FULL [OUTER] JOIN table2 USING (column1 [, ...]) |  
 CROSS JOIN table2 }  
WHERE  ...  
GROUP BY column_name | expr  
HAVING condition  
ORDER BY 기준1 [ ASC | DESC] [, 기준2 [ASC | DESC], ... ];
```

## JOIN - OUTER JOIN

조건을 만족시키지 못하는 행까지 Result Set에 포함시키는 조인 유형

EMP_NAME	DEPT_ID
한선기	90
강중훈	90
최만식	90
정도연	60
안석규	60
조재형	60
정지현	50
김예수	50
나승원	50
김순이	50
성해교	50
전우성	80
엄정하	80
심하균	
고승우	10
박하일	50
권상후	10
임영애	10
엄정하	
김솔오	20
이중기	20
감우섭	20

DEPT_ID	DEPT_NAME
20	회계팀
10	본사 인사팀
50	해외영업1팀
60	기술지원팀
80	해외영업2팀
90	해외영업3팀
30	마케팅팀

EMP_NAME	DEPT_NAME
한선기	해외영업3팀
강중훈	해외영업3팀
최만식	해외영업3팀
정도연	기술지원팀
안석규	기술지원팀
조재형	기술지원팀
정지현	해외영업1팀
김예수	해외영업1팀
나승원	해외영업1팀
김순이	해외영업1팀
성해교	해외영업1팀
전우성	해외영업2팀
엄정하	해외영업2팀
고승우	본사 인사팀
박하일	해외영업1팀
권상후	본사 인사팀
임영애	본사 인사팀
김솔오	회계팀
이중기	회계팀
감우섭	회계팀

EMP_NAME	DEPT_NAME
감우섭	회계팀
이중기	회계팀
김솔오	회계팀
임영애	본사 인사팀
권상후	본사 인사팀
고승우	본사 인사팀
박하일	해외영업1팀
성해교	해외영업1팀
김순이	해외영업1팀
나승원	해외영업1팀
김예수	해외영업1팀
정지현	해외영업1팀
조재형	기술지원팀
안석규	기술지원팀
정도연	기술지원팀
엄정하	해외영업2팀
전우성	해외영업2팀
최만식	해외영업3팀
강중훈	해외영업3팀
한선기	해외영업3팀
엄정하	
심하균	

## JOIN - OUTER JOIN <sup>1)</sup>오라클 전용 구문

- 연산자 '+' 사용
- 조인 조건을 만족시키는 행이 없는 테이블 기준

```
SELECT EMP_NAME, DEPT_NAME
FROM   EMPLOYEE E, DEPARTMENT D
WHERE  E.DEPT_ID = D.DEPT_ID(+);
```

소속 부서가 없는 직원까지 포함하는 의미

EMP_NAME	DEPT_NAME
감우섭	회계팀
이중기	회계팀
김술오	회계팀
임영애	본사 인사팀
권상후	본사 인사팀
고승우	본사 인사팀
박하일	해외영업1팀
성해교	해외영업1팀
김순이	해외영업1팀
나승원	해외영업1팀
김예수	해외영업1팀
정지현	해외영업1팀
조재형	기술지원팀
안석규	기술지원팀
정도연	기술지원팀
엄정하	해외영업2팀
전우성	해외영업2팀
최만식	해외영업3팀
강중훈	해외영업3팀
한선기	해외영업3팀
엄정하	
심하균	

EMP_NAME	DEPT_ID
한선기	90
강중훈	90
최만식	90
정도연	60
안석규	60
조재형	60
정지현	50
김예수	50
나승원	50
김순이	50
성해교	50
전우성	80
엄정하	80
심하균	
고승우	10
박하일	50
권상후	10
임영애	10
엄정하	
김술오	20
이중기	20
감우섭	20

DEPT_ID	DEPT_NAME
20	회계팀
10	본사 인사팀
50	해외영업1팀
60	기술지원팀
80	해외영업2팀
90	해외영업3팀
30	마케팅팀

--	--

## JOIN - OUTER JOIN <sup>1)</sup>오라클 전용 구문

```
SELECT EMP_NAME, DEPT_NAME
FROM   EMPLOYEE E, DEPARTMENT D
WHERE  D.DEPT_ID = E.DEPT_ID(+);
```

소속 직원이 없는 부서까지 포함하는 의미

EMP_NAME	DEPT_NAME
한선기	해외영업3팀
강중훈	해외영업3팀
최만식	해외영업3팀
정도연	기술지원팀
안석규	기술지원팀
조재형	기술지원팀
정지현	해외영업1팀
김예수	해외영업1팀
나승원	해외영업1팀
김순이	해외영업1팀
성해교	해외영업1팀
전우성	해외영업2팀
엄정하	해외영업2팀
고승우	본사 인사팀
박하일	해외영업1팀
권상후	본사 인사팀
임영애	본사 인사팀
김술오	회계팀
이중기	회계팀
감우섭	회계팀
	마케팅팀

EMP_NAME	DEPT_ID
한선기	90
강중훈	90
최만식	90
정도연	60
안석규	60
조재형	60
정지현	50
김예수	50
나승원	50
김순이	50
성해교	50
전우성	80
엄정하	80
심하균	
고승우	10
박하일	50
권상후	10
임영애	10
엄정하	
김술오	20
이중기	20
감우섭	20
	30

DEPT_ID	DEPT_NAME
20	회계팀
10	본사 인사팀
50	해외영업1팀
60	기술지원팀
80	해외영업2팀
90	해외영업3팀
30	마케팅팀

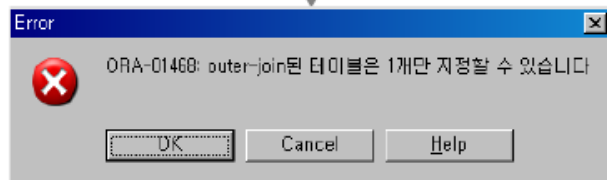


## JOIN - FULL OUTER JOIN

- 양쪽 테이블을 동시에 OUTER JOIN하는 ANSI 표준 구문
- 오라클 전용 구문은 지원되지 않음

```
SELECT EMP_NAME, DEPT_NAME  
FROM   EMPLOYEE  
FULL   JOIN DEPARTMENT USING (DEPT_ID);
```

```
SELECT EMP_NAME, DEPT_NAME  
FROM   EMPLOYEE E, DEPARTMENT D  
WHERE  E.DEPT_ID(+) = D.DEPT_ID(+);
```



EMP_NAME	DEPT_NAME
감우섭	회계팀
이중기	회계팀
김술오	회계팀
임영애	본사 인사팀
권상후	본사 인사팀
고승우	본사 인사팀
박하일	해외영업1팀
성해교	해외영업1팀
김순이	해외영업1팀
나승원	해외영업1팀
김예수	해외영업1팀
정지현	해외영업1팀
조재형	기술지원팀
안석규	기술지원팀
정도연	기술지원팀
엄정하	해외영업2팀
전우성	해외영업2팀
최만식	해외영업3팀
강중훈	해외영업3팀
한선기	해외영업3팀
엄정하	
심하균	
	마케팅팀