



SQL,PL/SQL(ORACLE)

3-1 조인

[조인(Join)]

1. 설명

하나의 테이블로는 원하는 컬럼정보를 참조할 수 없는 경우 관련된 테이블을 논리적으로 결합하여 원하는 컬럼정보를 참조하는 방법을 조인이라 한다.

2. 형식

논리적으로 결합되는 2개 이상의 테이블에는 반드시 '공통 컬럼'이 존재해야하며 이 '공통 컬럼'은 동일한 데이터 타입과 공통된 데이터를 포함해야 한다.

[형식 1. 오라클 조인]

```
SELECT 컬럼1, 컬럼2, 컬럼3...
FROM 테이블1, 테이블2
WHERE 테이블1.컬럼=테이블2.컬럼;
```

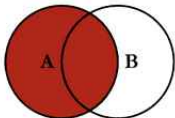
[형식 2. 표준(ANSI) 조인]

```
SELECT 컬럼1, 컬럼2, 컬럼3...
FROM 테이블1 join 테이블2
on 테이블1.컬럼=테이블2.컬럼;
```

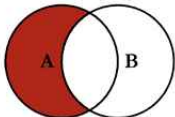
오라클 조인은 FROM절에 연결할 테이블들은 콤마로 구분하며, WHERE 절에서는 조인되는 테이블간의 공통 컬럼과 조인 조건을 적어준다.



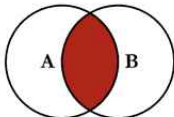
SQL JOINS



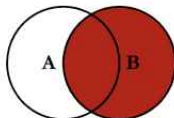
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



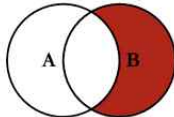
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



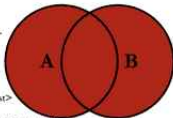
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



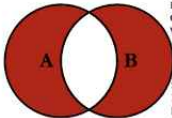
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

이름	널	유형
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

이름	널	유형
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

3. 두 테이블에 일치하는 컬럼값으로 조인(EQUI JOIN,이퀄조인,등가조인)

- 1) 사원테이블과 부서테이블을 조인해서 사원아이디와 부서명을 출력하라.

```
SELECT employees.employee_id 사원번호,
       departments.department_name 부서명
FROM employees, departments
WHERE employees.department_id=departments.department_id;
```

- 2) 사원테이블과 부서테이블을 조인해서 사원ID가 100인 사람의 사원ID와 근무지ID를 출력해라.

```
SELECT employees.employee_id 사원ID, departments.location_id 근무지ID
FROM employees, departments
WHERE employees.department_id=departments.department_id
AND employees.employee_id=100;
```

이름	널	유형
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

이름	널	유형
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

4. 테이블에 별칭 부여

- 1) 사원테이블과 부서테이블을 조인해서 사원번호와 부서명을 출력하라.

```
SELECT e.employee_id 사원번호, d.department_name 부서명
FROM employees e, departments d
WHERE e.department_id=d.department_id;
```

- 2) 사원테이블과 부서테이블을 조인해서 사원ID가 100인 사람의 사원ID와 근무지ID를 출력해라.

```
SELECT e.employee_id 사원ID, d.location_id 근무지ID
FROM employees e, departments d
WHERE e.department_id=d.department_id
AND e.employee_id=100;
```



이름	널	유형
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

이름	널	유형
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

5. 표준 조인(ANSI JOIN : 새로운 국제 표준, Oracle 9i버전에서부터 지원)

- 1) 사원테이블과 부서테이블을 조인해서 사원번호와 부서명을 출력하라.

```
select e.first_name 사원번호, d.department_name 부서명
from employees e join departments d
on( e.department_id = d.department_id);
```

- 2) 사원테이블과 부서테이블을 조인해서 사원ID가 100인 사람의 사원ID와 근무지ID를 출력해라.

```
SELECT e.employee_id 사원ID, d.location_id 근무지ID
FROM employees e JOIN departments d
ON e.department_id=d.department_id AND e.employee_id=100 ;
```



이름	널	유형
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

이름	널	유형
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)
JOB_ID	NOT NULL	VARCHAR2(10)
JOB_TITLE	NOT NULL	VARCHAR2(35)
MIN_SALARY		NUMBER(6)
MAX_SALARY		NUMBER(6)

6. 세 테이블일때도 가능 일치하는 컬럼값으로 조인(EQUI JOIN)

- 1)사원테이블과 부서테이블,업무테이블을 조인해서 사원번호와 부서명,업무제목을 출력하라.

```
SELECT e.employee_id 사원번호, d.department_name 근무부서,
       j.job_title 업무제목
```

```
FROM employees e, departments d, jobs j
WHERE e.department_id=d.department_id
AND e.job_id = j.job_id;
```

- 2)사원테이블과 부서테이블,업무테이블을 조인해서 사원ID가 100인 사람의 사원ID와 근무지ID,업무제목을 출력해라.

```
SELECT e.employee_id 사원ID, d.location_id 근무지ID, j.job_title 업무제목
FROM employees e, departments d, jobs j
WHERE e.department_id = d.department_id
AND j.job_id = e.job_id
AND e.employee_id=100;
```



이름	널	유형
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

이름	널	유형
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)
JOB_ID		NUMBER(4)
JOB_TITLE	NOT NULL	VARCHAR2(35)
MIN_SALARY		NUMBER(6)
MAX_SALARY		NUMBER(6)

7. 세 테이블일때도 표준 조인(ANSI JOIN)

- 1)사원테이블과 부서테이블,업무테이블을 조인해서 사원명과 부서명,업무제목을 출력하라.

```
select e.first_name 사원명, d.department_name 부서명,j.job_title 업무제목
from employees e join departments d
on e.department_id = d.department_id
join jobs j
on j.job_id = e.job_id;
```

- 2)사원테이블과 부서테이블,업무테이블을 조인해서 사원ID가 100인 사람의 사원ID와 근무지ID, 업무제목을 출력해라.

```
SELECT e.employee_id 사원ID, d.location_id 근무지ID,j.job_title 업무제목
FROM employees e JOIN departments d
ON e.department_id=d.department_id AND e.employee_id=100
JOIN jobs j
ON j.job_id = e.job_id;
```


Q1) employees 와 departments 테이블을 조인하여 사원이름이 'Steven'인 사원의 이름과 성, 부서명을 출력하되 부서명이 Executive일때는 행정부, Shipping일때는 발송부라고 출력하라.

```
SELECT e.first_name 이름, e.last_name 성,  
       decode(d.department_name,'Executive','행정부','Shipping','발송부') 부서명  
FROM employees e, departments d  
WHERE e.department_id=d.department_id AND e.first_name='Steven';
```

Q2) employees 와 departments 테이블을 조인하여 급여가 12000이상인 사원의 부서ID,부서명,이름,급여를 출력하라.(급여의 내림차순으로 정렬)

```
SELECT e.department_id 부서ID,d.department_name 부서명,  
       e.first_name 이름,e.last_name 성, e.salary 급여  
FROM employees e, departments d  
WHERE e.department_id=d.department_id AND e.salary>=12000  
ORDER BY e.salary DESC;
```



8. CROSS JOIN (Cartesian Product 카티션 곱)

2개 이상의 테이블이 조인될 때 where 절에 의해 공통되는 컬럼에 의한 결합이 발생하지 않는 경우 조인조건이 없으므로 두 테이블간의 조합가능한 모든 경우의 수를 계산하여 결과를 산출한다.

```
select count(*) from employees; //107행
```

```
select count(*) from jobs; //19행
```

```
select (107) * (19) from dual; //2033행
```

```
select  (select count(*) from employees)
        * (select count(*) from jobs) from dual; //2033행
```

```
select e.employee_id, e.salary, j.job_title
from employees e, jobs j; //2033행
```

표준조인(ANSI JOIN)의 경우 다음과 같이 작성.

```
SELECT e.employee_id, e.salary, j.job_title
FROM employees e cross join jobs j; //2033행
```



9. NON-EQUI JOIN (비등가 조인, 난 이퀄 조인)

WHERE절 또는 ON절에 사용된 '공통된 컬럼'들이 동등 연산자 (equals, =)에 의해 비교할 수 없는 즉 >, >=, <, <=, <>, BETWEEN ... AND 로 비교하는 경우에 사용되는 조인을 의미한다.

적당한 예시를 위해서 테이블(sal_grade)을 하나 생성하겠습니다.

HR_STUDY SAL_GRADE - Table x




Table Name: SAL_GRADE

Charset/Collation: Default Charset Default Collation

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
GRADE	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MIN_SAL	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MAX_SAL	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

- 생성한 급여등급테이블(SAL_GRADE)에
- 등급별 최소, 최대 급여를 오른쪽 그림과 같이 입력하겠습니다.

GRADE	MIN_SAL	MAX_SAL
6	15000	20000
5	12000	14999
4	10000	11999
3	8000	9999
2	5000	7999
1	2000	4999

- 30번 부서 사원의 급여목록입니다.

```
SELECT employee_id, salary
FROM employees
WHERE department_id = 30;
```

employee_id	salary
114	11000.00
115	3100.00
116	2900.00
117	2800.00

- 30번부서 직원들의 급여목록에 급여등급테이블을 조인해서 등급을 부여하려 합니다.
- 각 사원의 급여가 등급별 최소급여이상 최대급여이하 범위에 따라 등급이 정해집니다.

```
SELECT e.employee_id, e.salary,
       s.max_sal, s.grade
FROM employees e JOIN sal_grade s
ON e.salary
BETWEEN s.min_sal AND s.max_sal
WHERE e.department_id = 30;
```

employee_id	salary	max_sal	grade
119	2500.00	4999	1
118	2600.00	4999	1
117	2800.00	4999	1
116	2900.00	4999	1
115	3100.00	4999	1
114	11000.00	11999	4



이름	널	유형
EMPLOYEE_ID	NOT NULL	NUMBER (6)
FIRST_NAME		VARCHAR2 (20)
LAST_NAME	NOT NULL	VARCHAR2 (25)
EMAIL	NOT NULL	VARCHAR2 (25)
PHONE_NUMBER		VARCHAR2 (20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2 (10)
SALARY		NUMBER (8, 2)
COMMISSION_PCT		NUMBER (2, 2)
MANAGER_ID		NUMBER (6)
DEPARTMENT_ID		NUMBER (4)

	EMPLOYEE_ID	E1.FIRST_NAME '의 매니저' E2.FIRST_NAME
1	101	Neena 의 매니저 Steven
2	102	Lex 의 매니저 Steven
3	103	Alexander 의 매니저 Lex
4	104	Bruce 의 매니저 Alexander
5	105	David 의 매니저 Alexander
6	106	Valli 의 매니저 Alexander
7	107	Diana 의 매니저 Alexander
8	108	Nancy 의 매니저 Neena
9	109	Daniel 의 매니저 Nancy
10	110	John 의 매니저 Nancy
11	111	Ismael 의 매니저 Nancy
12	112	Jose Manuel 의 매니저 Nancy
13	113	Luis 의 매니저 Nancy
14	114	Den 의 매니저 Steven
15	115	Alexander 의 매니저 Den

강사 양영재

10. SELF JOIN

참조해야 할 컬럼이 자신의 테이블에 있는 다른 컬럼인 경우 사용하는 조인 (단, NULL 일때는 결과에서 배제되어진다.Steven은 매니저가 null 이어서 employee_id 100번이 표시 안된 것을 알 수 있으며, nvl처리로 해결되지 않는다 >> 다음장에 배울 outer join으로 처리)

예)employees 테이블에서 자신의 매니저의 이름을 검색하세요.

```
SELECT e1.employee_id,e1.first_name || ' 의 매니저 ' || nvl(e2.first_name,'없음')
FROM employees e1,employees e2
WHERE e1.manager_id=e2.employee_id
ORDER BY e1.employee_id ASC;
```

EMPLOYEE_ID	MANAGER_ID	FIRST_NAME	LAST_NAME	T_NAME
100	1004	Steven		1004
101	1004	Neena		1004
102	1004	Lex		1004
103	1004	Alexander		1004
104	1004	Bruce		1004
105	1004	David		1004
106	1004	Valli		1004
107	1004	Diana		1004
108	1004	Nancy		1004
109	1004	Daniel		1004
110	1004	John		1004
111	1004	Ismael		1004
112	1004	Jose Manuel		1004
113	1004	Luis		1004
114	1004	Den		1004

EMPLOYEE_ID	FIRST_NAME	DEPARTMENT_NAME
(null)	(null)	Benefits
(null)	(null)	Recruiting
(null)	(null)	Retail Sales
(null)	(null)	Government Sales
(null)	(null)	IT Helpdesk
(null)	(null)	NOC
(null)	(null)	Treasury
(null)	(null)	Corporate Tax
(null)	(null)	Control And Credit
(null)	(null)	Shareholder Services
(null)	(null)	Payroll
(null)	(null)	Manufacturing
(null)	(null)	Construction
(null)	(null)	Contracting
(null)	(null)	Operations
(null)	(null)	IT Support
206	William	Accounting
205	Shelley	Accounting
204	Hermann	Public Relations
203	Susan	Human Resources

11. OUTER JOIN

1) 한쪽 테이블에는 해당하는 데이터가 존재하는 데 다른쪽 테이블에는 데이터가 존재하지 않을 경우 모든 데이터를 출력하게 하는 조인

(즉, NULL 값을 포함하더라도 출력하고자 할때)

```
SELECT e1.employee_id, e1.first_name || '의 매니저 ' || nvl(e2.first_name, 1004)
FROM employees e1, employees e2
WHERE e1.manager_id = e2.employee_id(+)
ORDER BY e1.employee_id ASC;
```

```
SELECT e.employee_id, e.first_name, d.department_name
FROM employees e, departments d
WHERE e.department_id(+) = d.department_id
ORDER BY e.employee_id DESC;
```



2) OUTER JOIN 의 특징

- (+)는 WHERE 절에서만 사용가능하다.

- 테이블간의 외부조인 조건이 한 개 이상일 경우, 모든 외부조인 조건에 (+)를 붙여야 외부조인이 성립된다. 또한 정확한 외부조건 결과를 얻기 위해서는 조인조건외의 일반조건에도 (+)를 붙여야 한다.

- (+)는 테이블자신에 붙을수 없으면 오로지 컬럼에만 붙는다.

- OR연산자와 같이 사용할 수 없다. AND연산은 가능.

예) `SELECT e.first_name, e.job_id, d.department_id, d.location_id
FROM employees e,departments d
WHERE e.department_id(+) = d.department_id AND job_id='SA_MAN';`

- 조인조건식에서 (+)가 붙은 컬럼과는 IN연산자와 같이 사용할 수 없다.

`SELECT e.first_name, e.job_id, d.department_id, d.location_id
FROM employees e,departments d
WHERE e.department_id(+) = d.department_id IN('SA_MAN','PU_CLERK');`

- 조인조건식에서 (+)가 붙은 컬럼과는 서브쿼리를 같이 사용할 수 없다.



12. ANSI 조인 : 새로운 국제 표준, Oracle 9i버전에서부터 지원

1) ANSI 내부조인 (inner join)

ex) 사원테이블과 부서테이블을 조인하여 사원ID, 사원이름, 부서명을 출력하라.
(사원ID오름차순 정렬)

- 기존방법

```
select e.employee_id 사원ID, e.first_name 사원이름,  
       d.department_name 부서명  
from employees e , departments d  
where e.department_id = d.department_id order by e.employee_id asc;
```

- WHERE절 대신 ON 사용

```
select e.employee_id 사원ID, e.first_name 사원이름,  
       d.department_name 부서명  
from employees e INNER JOIN departments d  
ON e.department_id = d.department_id order by e.employee_id asc;
```

- WHERE절 대신 USING 사용

```
select e.employee_id 사원ID, e.first_name 사원이름,  
       d.department_name 부서명  
from employees e INNER JOIN departments d  
USING(department_id) order by e.employee_id asc;
```



2) ANSI 외부조인 (outer join)

[형식] FROM 테이블명 [**LEFT** | **RIGHT** | **FULL**] OUTER JOIN 테이블명

ex) 사원테이블과 부서테이블을 조인하여 사원ID,사원이름, 부서명을 출력하라.
(사원ID오름차순 정렬)

```
select e.employee_id 사원ID, e.first_name 사원이름,  
       d.department_name 부서명  
from employees e LEFT OUTER JOIN departments d  
USING(department_id) order by e.employee_id asc;
```

```
select e.employee_id 사원ID, e.first_name 사원이름,  
       d.department_name 부서명  
from employees e RIGHT OUTER JOIN departments d  
USING(department_id) order by e.employee_id asc;
```

```
select e.employee_id 사원ID, e.first_name 사원이름,  
       d.department_name 부서명  
from employees e FULL OUTER JOIN departments d  
USING(department_id) order by e.employee_id asc;
```

- USING 대신에 ON 을 이용한 조건절도 가능



2) NATURAL JOIN

- 두 테이블의 컬럼명이 같은 경우 조인 컬럼을 자동으로 알아서 조인하므로 이러한 등가조인(Equi Join)을 자연조인(NATURAL JOIN)이라고 하고 ON절이나 USING문을 사용하지 않아도 됩니다.
- 반드시 두 테이블 간의 동일한 이름, 타입을 가진 컬럼이 필요하다.
- 조인에 이용되는 컬럼은 명시하지 않아도 자동으로 조인에 사용된다.
- 동일한 이름을 갖는 컬럼이 있지만 데이터 타입이 다르면 에러가 발생한다.
- 주의 : 편리해보이지만 성립요건이 까다로운만큼 INNER JOIN 사용권장하며
- 오류가 없더라도 INNER JOIN 결과와 다르게 출력될 수 있습니다.

```
SELECT employee_id, department_name
FROM employees NATURAL JOIN departments;
```

employee_id	department_name
202	Marketing
115	Purchasing

```
SELECT department_id, department_name,
       location_id, city
FROM departments NATURAL JOIN locations;
```

department_id	department_name	location_id	city
10	Administration	1700	Seattle
20	Marketing	1800	Toronto

1. 사원 테이블과 부서 테이블을 조인하여 모든 사원ID, 사원이름, 급여, 부서명을 출력하라. (부서명 내림차순 정렬)

```
SELECT e.employee_id 사원ID, e.first_name 사원이름,
       e.salary 급여, d.department_name 부서명
FROM employees e LEFT OUTER JOIN departments d
USING(department_id) order by d.department_name desc;
```

2. 사원 테이블과 부서 테이블을 조인하여 직업ID가 'IT_PROG' 인 사원들의 사원이름, 직업ID, 부서명, 위치ID를 출력하세요.

```
SELECT e.first_name 사원이름, e.job_id 직업ID, d.department_name 부서명,
       d.location_id 위치ID
FROM employees e INNER JOIN departments d
ON e.department_id=d.department_id AND job_id='IT_PROG';
```

3. 부서 테이블과 사원 테이블에서 사번, 사원명, 업무, 급여, 부서명을 검색하시오. 단, 업무명이 '%Manager' 이며 급여가 8000 이상인 사원에 대하여 사번을 기준으로 오름차순 정렬할 것.

```
select e.employee_id 사번, e.first_name 사원명, j.job_title 업무명,
       e.salary 급여, d.department_name 부서명
from employees e JOIN departments d ON e.department_id = d.department_id
JOIN jobs j ON j.job_id = e.job_id AND job_title LIKE '%Manager' AND
e.salary >= 8000
order by e.employee_id asc;
```