

6. 조인(JOIN)



(1) 표준조인

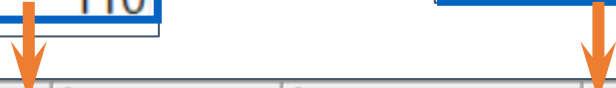
여러 테이블에서 데이터 얻기

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
...		
201	Hartstein	20
202	Fay	20
205	Higgins	110
206	Gietz	110

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700



EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
100	King	90	Executive
101	Kochhar	90	Executive
102	De Haan	90	Executive
103	Hunold	60	IT
...			
201	Hartstein	20	Marketing
202	Fay	20	Marketing
205	Higgins	110	Accounting
206	Gietz	110	Accounting

ANSI 조인 유형

- SQL:1999 표준과 호환되는 조인(ANSI JOIN)에는 다음이 포함됩니다
 - Natural join:
 - NATURAL JOIN 절
 - USING 절
 - ON 절
 - OUTER join:
 - LEFT OUTER JOIN
 - RIGHT OUTER JOIN
 - FULL OUTER JOIN
 - Cross join



조인 정의 및 구문

- 데이터베이스에서 여러 테이블의 데이터가 필요한 경우 조인 조건을 사용합니다.
- 서로 대응되는 열에 존재하는 공통 값에 따라 한 테이블의 행(row)을 다른 테이블의 행에 조인할 수 있습니다.
- 구문

```
SELECT table1.column, table2.column  
FROM   table1  
[NATURAL JOIN table2] |  
[JOIN table2 USING (column_name)] |  
[JOIN table2  
  ON(table1.column_name = table2.column_name)];
```

자연 조인 작성

- NATURAL JOIN 절은 두 테이블에서 동일한 이름을 가진 모든 열을 기준으로 합니다.
- 두 테이블의 일치하는 모든 열에서 같은 값을 가진 행을 선택합니다.
- 동일한 이름을 가진 열의 데이터 유형이 서로 다를 경우 오류가 반환됩니다.
- 다음 예제는 DEPARTMENTS와 LOCATIONS를 location_id열을 기준으로 조인하여 각 부서가 위치한 도시를 표시합니다.

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

USING 절을 포함하는 조인 작성

- 여러 열이 같은 이름을 가지지만 데이터 유형이 일치하지 않을 경우, NATURAL JOIN 절을 수정하여 USING 절을 포함시키면 등가 조인에 사용될 열을 지정할 수 있습니다.
- USING 절을 사용하면 둘 이상의 열이 일치할 때 한 열만 일치시킬 수 있습니다.
- 참조되는 열에 테이블 이름이나 별칭을 사용해서는 안됩니다.
- NATURAL JOIN과 USING 절은 함께 사용할 수 없습니다.

USING 절로 레코드 검색

- 예제는 EMPLOYEES 및 DEPARTMENTS 테이블의 DEPARTMENT_ID 열을 USING 절로 조인하여 사원이 일하는 부서이름을 함께 표시합니다.

```
SELECT employee_id, last_name, department_name  
FROM   employees JOIN departments  
USING (department_id) ;
```


ON 절로 조인 작성

- 자연 조인과 USING 절을 사용하는 조인은 기본적으로 같은 이름을 가진 모든 열의 등가 조인입니다.
- 임의의 조건을 지정하거나 조인할 열을 지정하려면 ON 절을 사용합니다.
- ON 절을 사용하면 조인 조건이 다른 검색 조건과 분리되므로 코드를 이해하기 쉽습니다.

ON 절을 사용하여 레코드 검색

- 다음은 ON절을 사용하는 JOIN문으로 사원이 근무하는 부서번호와 부서위치를 표시합니다.
- DEPARTMENT_ID 열이 두 테이블에 모두 있으므로 테이블 이름을 접두어로 사용하여 구분해야 합니다.

```
SELECT employees.employee_id, employees.last_name,  
       employees.department_id, departments.department_id,  
       departments.location_id  
FROM   employees JOIN departments  
ON     (employees.department_id = departments.department_id);
```

테이블 접두어의 사용

- 테이블 접두어를 사용하여 여러 테이블에 있는 모호한 열 이름을 한정합니다.
- 테이블 접두어를 사용하면 성능을 향상시킵니다.
- 전체 테이블 이름의 접두어 대신 테이블 Alias를 사용할 수 있습니다.
 - 테이블 Alias로 테이블에 짧은 이름을 지정하면 SQL 코드 크기를 줄여 메모리를 적게 사용하게 됩니다.
- 열 alias를 사용하여 이름은 같지만 서로 다른 테이블에 상주하는 열을 구분할 수 있습니다.

테이블 접두어의 사용 예제

- 예제는 EMPLOYEES 및 DEPARTMENTS 테이블의 DEPARTMENT_ID 열을 조인하여 사원이 일하는 부서 표시합니다.

```
SELECT employees.employee_id, employees.last_name,  
       departments.department_name  
FROM   employees JOIN departments  
ON     (employees.department_id = departments.department_id)
```

- 접두어로 테이블 이름 대신 테이블 Alias를 사용할 수 있습니다.

```
SELECT e.employee_id, e.last_name,  
       d.department_name  
FROM   employees e JOIN departments d  
ON     (e.department_id = d.department_id);
```

ON 절로 3-way 조인 작성

- 3-way 조인은 세 테이블 조인입니다
- 3-Way 조인으로 사원이 근무하는 부서 및 부서가 위치하는 도시를 출력할 수 있습니다.

```
SELECT employee_id, city, department_name
FROM   employees e
JOIN   departments d
ON     (d.department_id = e.department_id)
JOIN   locations l
ON     (d.location_id = l.location_id);
```

조인에 추가 조건 적용

- AND 또는 WHERE 절을 사용하여 추가 조건을 적용합니다.

```
SELECT e.employee_id, e.last_name, e.department_id,  
d.department_id, d.location_id  
FROM employees e JOIN departments d  
ON (e.department_id = d.department_id)  
AND e.manager_id = 149 ;
```

```
SELECT e.employee_id, e.last_name, e.department_id,  
d.department_id, d.location_id  
FROM employees e JOIN departments d  
ON (e.department_id = d.department_id)  
WHERE e.manager_id = 149 ;
```

Self Join

- 하나의 테이블을 두 번 사용하여 조인을 수행할 수 있습니다.
- Self Join문은 테이블 접두어 또는 별칭 사용이 필수입니다.
- 다음은 사원테이블을 Self Join하여 각 사원의 관리자 이름을 표시합니다.

```
SELECT worker.last_name emp, manager.last_name mgr  
FROM employees worker JOIN employees manager  
ON (worker.manager_id = manager.employee_id)
```

Non-equi Join

- 등호연산자(=)를 사용하거나 USING절을 사용한 JOIN 또는 NATURAL JOIN은 모두 Equi Join입니다.
- Non-equi join은 ON절에 등호 연산자(=)가 아닌, 다른 연산자를 포함하는 조인입니다.
- 다음은 사원테이블과 JOB_GRADES 테이블을 BETWEEN 연산자가 사용된 Non-equi Join으로 각 사원의 급여를 기준으로 등급을 표시합니다.

```
SELECT e.last_name, e.salary, j.grade_level  
FROM employees e JOIN job_grades j  
ON e.salary BETWEEN j.lowest_sal AND j.highest_sal;
```


내부 조인(inner join)과 포괄 조인(outer join)

- 조인 조건을 충족하지 못하는 행은 결과에 나타나지 않는 조인 조건을 충족하는 행만 결과로 출력하는 조인이 내부 조인입니다.
 - 두 테이블을 조인해서 내부 조인의 결과와 함께 일치하지 않는 왼쪽 또는 오른쪽 테이블의 행을 반환하는 조인이 Left 또는 Right Outer Join입니다.
 - 두 테이블을 조인해서 내부 조인의 결과와 함께 모든 행을 반환하는 조인이 Full Outer Join입니다.
- 구문

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

LEFT / RIGHT OUTER JOIN

- 이 예제는 왼쪽 테이블인 EMPLOYEES 테이블의 부서가 없는 사원을 포함한 모든 행을 검색합니다.

```
SELECT e.last_name, e.department_id, d.department_name  
FROM   employees e LEFT OUTER JOIN departments d  
ON     (e.department_id = d.department_id) ;
```

- 이 예제는 오른쪽 테이블인 DEPARTMENTS 테이블의 사원이 없는 빈 부서를 포함한 모든 행을 검색합니다.

```
SELECT e.last_name, e.department_id, d.department_name  
FROM   employees e RIGHT OUTER JOIN departments d  
ON     (e.department_id = d.department_id) ;
```

FULL OUTER JOIN

- 다음 예제는 Inner Join의 결과와 함께 부서가 정해지지 않은 사원 및 사원이 없는 부서를 모두 포함하여 검색합니다.

```
SELECT e.last_name, e.department_id, d.department_name  
FROM   employees e FULL OUTER JOIN departments d  
ON     (e.department_id = d.department_id) ;
```

카티시안 곱(Cartesian Product)

- 카티시안 곱은 다음 경우에 생성됩니다.
 - 조인 조건을 생략한 경우
 - 조인 조건이 부적합한 경우
 - 첫번째 테이블의 모든 행이 두번째 테이블의 모든 행에 조인된 경우
- 카티시안 곱이 생성되지 않도록 하려면 ON절에 항상 유효한 조인 조건을 포함시켜야 합니다.

카티시안 곱(Cartesian Product)

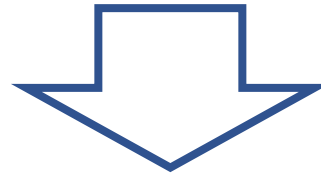
EMPLOYEES (20개 행)

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
107	Lorentz	60
124	Mourgos	50
141	Rajs	50
142	Davies	50

...

DEPARTMENTS (8개 행)

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700



카티시안 곱 : $20 \times 8 = 160$ 개 행

카티시안 곱(Cartesian Product) 예제

- EMPLOYEES 와 LOCATIONS를 자연조인 하면, 조인키 속성이 없어서 카티시안 곱을 결과로 생성합니다.

```
SELECT last_name, city  
FROM employees NATURAL JOIN locations;
```

Cross Join 작성

- CROSS JOIN 절은 두 테이블 상호간의 조합을 생성합니다.
- 이것은 두 테이블 사이의 카티시안 곱(Cartesian Product)과 동일합니다.
- 구문

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

- 예문

```
SELECT last_name, department_name  
FROM   employees CROSS JOIN departments ;
```

(2) 비표준조인

조인의 종류

- 등가조인(Equi Join)
- 비등가조인(Non Equi Join)
- 내부조인(Inner Join)
- 외부조인(Outer Join)
- 자체조인(Self Join)
- 3-Way Join



조인 조건에 사용된 연산자



조인조건을 충족하지 않는 행의 출력 여부



조인에 참여하는 테이블의 개수

비표준 조인

- FROM 절에 JOIN 키워드를 사용하지 않습니다.
- 조인조건을 WHERE 절에 작성합니다.
 - n개의 테이블 조인을 위해 최소 n-1개의 조인 조건이 필요합니다.
- 행을 제한하기 위해 WHERE 절에 조건을 추가해야하는 경우 AND 연산자를 사용합니다.
- 구문

```
SELECT table1.column, table2.column  
FROM   table1, table2  
WHERE  table1.column1 = table2.column2;
```

등가조인

- 예제는 다음 사항을 조인을 통해 함께 출력합니다.
 - EMPLOYEES 테이블에 있는 사원의 이름, 사원 번호 및 부서 번호 열
 - DEPARTMENTS 테이블에 있는 부서 번호, 부서 이름 및 위치 ID 열
- WHERE 절에 부서번호 50과 60번을 추가조건으로 지정하여 해당 부서 사원으로 결과를 제한합니다.

```
SELECT e.employee_id, e.last_name, e.department_id,  
       d.department_name, d.location_id  
FROM   employees e , departments d  
WHERE  e.department_id = d.department_id  
AND    e.department_id IN (50,60);
```

비등가조인과 셀프조인

- 다음은 사원테이블과 JOB_GRADES 테이블을 BETWEEN 연산자가 사용된 Non-equi Join으로 각 사원의 급여를 기준으로 등급을 표시합니다.

```
SELECT e.last_name, e.salary, j.grade_level  
FROM employees e, job_grades j  
WHERE e.salary BETWEEN j.lowest_sal AND j.highest_sal;
```

- 다음은 EMPLOYEES 테이블을 두 번 검색하여 사원의 매니저 이름을 반환하는 셀프조인입니다.

```
SELECT e.employee_id, e.last_name, e.manager_id,  
       m.last_name as manager_name  
FROM employees e, employees m  
WHERE e.manager_id = m.employee_id;
```

외부조인(Outer Join)

- 외부조인은 아우터조인 연산자를 정보가 부족한 조인 “옆”에 넣습니다.
 - 아우터 조인 연산자는 더하기 기호를 괄호로 묶어(+) 표시합니다.
- 아우터 조인 연산자는 하나 이상의 널 행을 생성합니다.
- 구문

```
SELECT table1.column, table2.column  
FROM   table1, table2  
WHERE  table1.column1(+) = table2.column2;
```

OR

```
SELECT table1.column, table2.column  
FROM   table1, table2  
WHERE  table1.column1 = table2.column2(+);
```

외부조인 예제

- 예제는 사원이 없는 부서정보를 포함하여 사원의 이름, 부서 ID 및 부서 이름을 표시합니다.
- Contracting 부서에는 사원이 없으므로 출력 결과에 공백 값이 표시됩니다

```
SELECT e.last_name, e.department_id, d.department_name  
FROM   employees e, departments d  
WHERE  e.department_id(+) = d.department_id ;
```

- 이 방식은 표준조인의 RIGHT OUTER JOIN과 동일합니다.

외부조인 예제

- 이 예제는 왼쪽 테이블인 EMPLOYEES 테이블의 부서가 없는 사원을 포함한 모든 행을 검색합니다.
- Grant 사원의 부서이름은 Null 입니다.

```
SELECT e.last_name, e.department_id, d.department_name  
FROM   employees e, departments d  
WHERE  e.department_id = d.department_id (+) ;
```

- 이 방식은 표준조인의 LEFT OUTER JOIN과 동일합니다.

외부 조인 고려사항

- 외부조인 연산자(+)는 한 문장에 한 번만 사용해야 합니다.
- 다음 예제는 오류가 발생합니다.

```
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e, departments d
WHERE  e.department_id(+) = d.department_id
AND    e.department_id = d.department_id+);
```

ORA-01416: two tables cannot be outer-joined to each other

Full Outer Join

- 오라클은 다음과 같이 집합연산자 UNION을 사용하여 Full Outer Join의 결과를 출력할 수 있습니다.
- 다음 예제는 사원이 없는 부서와 부서가 정해지지 않은 사원이 모두 출력됩니다.

```
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e, departments d
WHERE  e.department_id(+) = d.department_id
UNION
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e, departments d
WHERE  e.department_id = d.department_id(+);
```

Thank you 😊