

JOIN 이란? ANSI JOIN



두 개의 테이블을 서로 연관해서 조회하는 것을 조인이라 부릅니다

EMPLOYEES

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_ID
100	Steven	King	90
101	Neena	Kochhar	90
102	Lex	De Haan	90
103	Alexander	Hunold	60
104	Bruce	Ernst	60
105	David	Austin	60
106	Valli	Pataballa	60
107	Diana	Lorentz	60
108	Nancy	Greenberg	100
109	Daniel	Faviet	100
110	John	Chen	100
111	Ismael	Sciarra	100
112	Jose Manuel	Urman	100
113	Luis	Popp	100
114	Den	Raphaely	30
115	Alexander	Khoo	30
116	Shelli	Baida	30

DEPARTMENTS

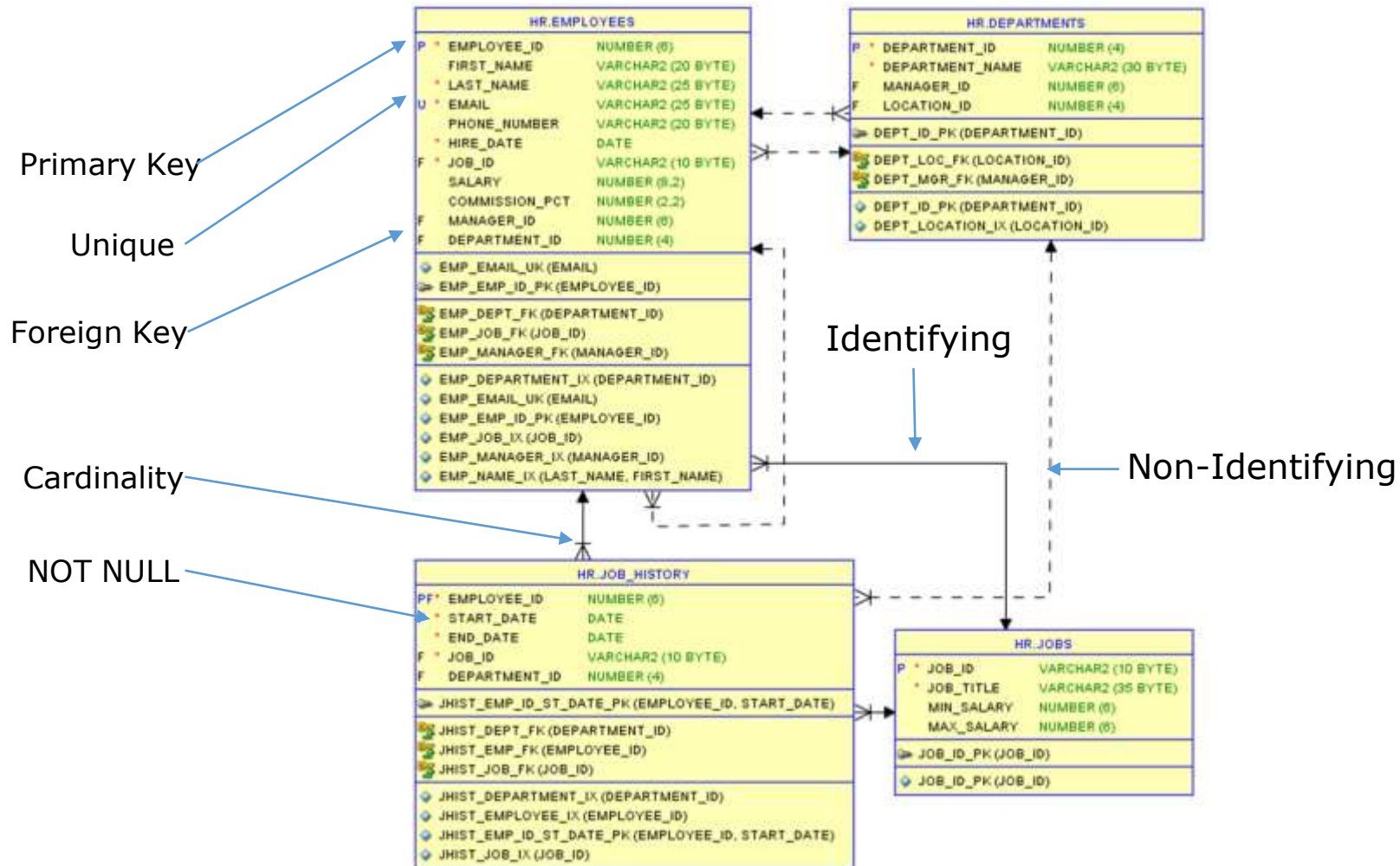
DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing	114	1700
40	Human Resources	203	2400
50	Shipping	121	1500
60	IT	103	1400
70	Public Relations	204	2700
80	Sales	145	2500
90	Executive	100	1700
100	Finance	108	1700
110	Accounting	205	1700
120	Treasury	(null)	1700
130	Corporate Tax	(null)	1700



Join 결과

FIRST_NAME	DEPARTMENT_NAME
Jennifer	Administration
Pat	Marketing
Michael	Marketing
Sigal	Purchasing
Karen	Purchasing
Shelli	Purchasing
Den	Purchasing
Alexander	Purchasing
Guy	Purchasing
Susan	Human Resources

- 하나 이상의 테이블로부터 데이터를 질의하기 위해서 조인을 사용합니다.
- 하나 이상의 테이블에 똑같은 열 이름이 있을 때 열 이름 앞에 테이블 이름을 붙입니다.
- 오라클 조인 구문과 **ANSI 조인 구문**이 있습니다.



중요 조인의 종류

1. INNER JOIN - 내부 조인(EQUALS JOIN)

2. OUTER JOIN - 외부 조인

-LEFT

-RIGHT

-FULL OUTER

그 밖의 조인의 종류

3. CLOSS JOIN

4. SELF JOIN

- 오라클 9i 버전부터 ANSI SQL 표준 문법 사용이 가능합니다.
- 조인의 형태가 FROM 절에서 지정되며, 조인 조건이 ON 절 또는 USING 절에 표시됩니다.

```
SELECT  table1.column, table2.column
FROM    table1
[LEFT | RIGHT | FULL] [OUTER] JOIN  table2
ON      join_conditions
[WHERE  conditions]
```


구문에서...

- ON *join_conditions* : ON 절에 조인 조건을 작성합니다. 조인 조건에 따라 Equi 조인 또는 Non-Equi 조인이 될 수 있습니다.

INNER JOIN

JOIN구문 앞에 INNER를 생략 할 수 있습니다

ON 절을 이용하면 JOIN 이후에 논리 연산과 서브쿼리와 같은 추가 서술을 할 수 있습니다.



```
SQL> SELECT department_name, street_address, city, state_province
2 FROM departments d
3 JOIN locations l
4 ON d.location_id=l.location_id;
```

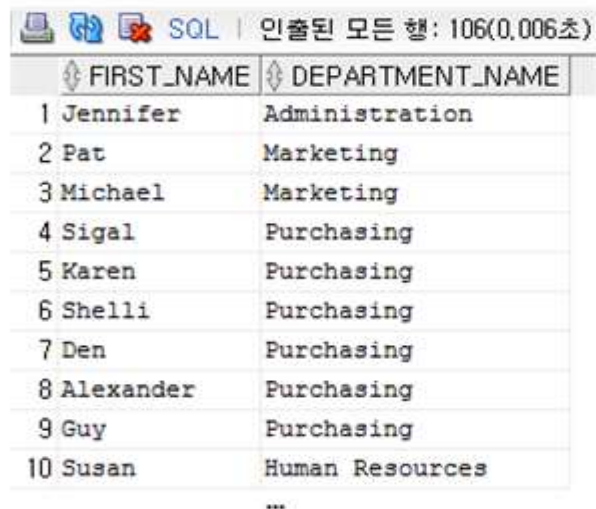
SQL | 인출된 모든 행: 27(0.003초)

DEPARTMENT_NAME	STREET_ADDRESS	CITY	STATE_PROVINCE
1 IT	2014 Jabberwocky Rd	Southlake	Texas
2 Shipping	2011 Interiors Blvd	South San Francisco	California
3 Administration	2004 Charade Rd	Seattle	Washington
4 Purchasing	2004 Charade Rd	Seattle	Washington
5 Executive	2004 Charade Rd	Seattle	Washington
6 Finance	2004 Charade Rd	Seattle	Washington
7 Accounting	2004 Charade Rd	Seattle	Washington
8 Treasury	2004 Charade Rd	Seattle	Washington
9 Corporate Tax	2004 Charade Rd	Seattle	Washington
10 Control And Credit	2004 Charade Rd	Seattle	Washington

...

USING 절을 이용하면 원하는 컬럼에 대해서만 선택적으로 INNER 조인을 할 수 있습니다.

```
SQL> SELECT first_name, department_name  
2 FROM employees  
3 JOIN departments  
4 USING (department_id);
```

 SQL | 인출된 모든 행: 106(0.006초)

FIRST_NAME	DEPARTMENT_NAME
1 Jennifer	Administration
2 Pat	Marketing
3 Michael	Marketing
4 Sigal	Purchasing
5 Karen	Purchasing
6 Shelli	Purchasing
7 Den	Purchasing
8 Alexander	Purchasing
9 Guy	Purchasing
10 Susan	Human Resources
...	

JOIN~ON 절을 반복적으로 사용하면 3개 이상 테이블에서 조인을 수행할 수 있습니다.

```
SQL> SELECT e.first_name, d.department_name,
2      l.street_address || ', ' || l.city || ', ' || l.state_province
3      AS address
4 FROM employees e
5 JOIN departments d ON e.department_id=d.department_id
6 JOIN locations l ON d.location_id=l.location_id;
```

   SQL | 인출된 모든 행: 106(0.006초)

	FIRST_NAME	DEPARTMENT_NAME	ADDRESS
1	Ellen	Sales	Magdalen Centre, The Oxford Science Park, Oxford, Oxford
2	Sundar	Sales	Magdalen Centre, The Oxford Science Park, Oxford, Oxford
3	Mozhe	Shipping	2011 Interiors Blvd, South San Francisco, California
4	David	IT	2014 Jabberwocky Rd, Southlake, Texas
5	Hermann	Public Relations	Schwanthalerstr. 7031, Munich, Bavaria
6	Shelli	Purchasing	2004 Charade Rd, Seattle, Washington
7	Amit	Sales	Magdalen Centre, The Oxford Science Park, Oxford, Oxford
8	Elizabeth	Sales	Magdalen Centre, The Oxford Science Park, Oxford, Oxford
9	Sarah	Shipping	2011 Interiors Blvd, South San Francisco, California
10	David	Sales	Magdalen Centre, The Oxford Science Park, Oxford, Oxford

ANSI 조인은 조인 조건과 WHERE 절 조건을 분리해서 작성할 수 있으므로 쿼리문을 직관적으로 이해하기 쉽습니다.

```
SQL> SELECT e.first_name AS name,
2         d.department_name AS department
3 FROM   employees e
4 JOIN   departments d
5 ON     e.department_id=d.department_id
6 WHERE  employee_id = 103;
```

NAME	DEPARTMENT
1 Alexander	IT

다음 구문은 103번 사원의 이름과, 부서이름 그리고 주소를 출력합니다.

```
SQL> SELECT e.first_name AS name, d.department_name AS department,
2         l.street_address || ', ' || l.city || ', ' || l.state_province
3         AS address
4 FROM   employees e
5 JOIN   departments d
6 ON     e.department_id=d.department_id
7 JOIN   locations l
8 ON     d.location_id=l.location_id
9 WHERE  employee_id = 103;
```

NAME	DEPARTMENT	ADDRESS
1 Alexander	IT	2014 Jabberwocky Rd, Southlake, Texas

조인문장을 작성할 때 일반 조건을 반드시 WHERE 절에 기술하지 않아도 됩니다. 조인 문장의 ON 절에 일반 조건을 포함시킬 수 있습니다.

다음 코드는 103번 사원의 이름과 부서이름을 출력합니다.

```
SQL> SELECT e.first_name AS name,  
2         d.department_name AS department  
3 FROM   employees e  
4 JOIN   departments d  
5 ON     e.department_id=d.department_id AND employee_id = 103;
```

	NAME	DEPARTMENT
1	Alexander	IT

OUTER JOIN

- LEFT OUTER
- RIGHT OUTER
- FULL OUTER

지금까지 오라클에서 제공하였던 Outer 조인 표기는 (+) 문자를 사용했었습니다. 오라클 Outer 조인은 이해가 어렵고 실수를 유발하기 쉽다는 단점이 있었습니다. ANSI SQL-3의 기준을 오라클 9i에서 수용하면서 LEFT / RIGHT OUTER 조인뿐만 아니라, 그 동안 UNION이나 UNION ALL을 이용해서 처리하던 양쪽 Outer 조인도 FULL OUTER JOIN 문법으로 새로 추가 되었습니다.

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

LEFT OUTER JOIN

다음 구문은 모든 사원의 정보를 출력하면서 직무 변동 기록을 같이 출력합니다. Outer 조인을 수행할 때 해당 데이터를 모두 가져올 테이블의 방향을 기록합니다. 다음 구문은 모든 사원에 대해 직무 이력이 없는 정보도 출력해야 하므로 Left Outer 조인을 사용했습니다.

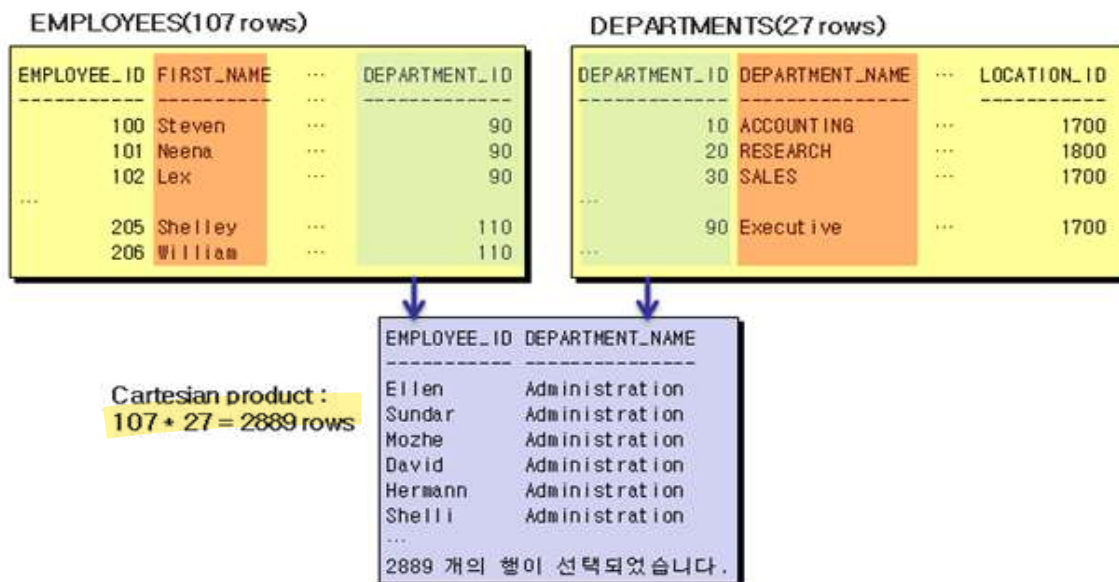
```
SQL> SELECT e.employee_id, e.first_name, e.hire_date,
2         j.start_date, j.end_date, j.job_id, j.department_id
3 FROM   employees e
4 LEFT OUTER JOIN job_history j
5 ON     e.employee_id = j.employee_id
6 ORDER BY j.employee_id;
```

위 구문을 실행하면 110개 행이 출력됩니다.

   SQL | 인출된 모든 행: 110(0.005초)

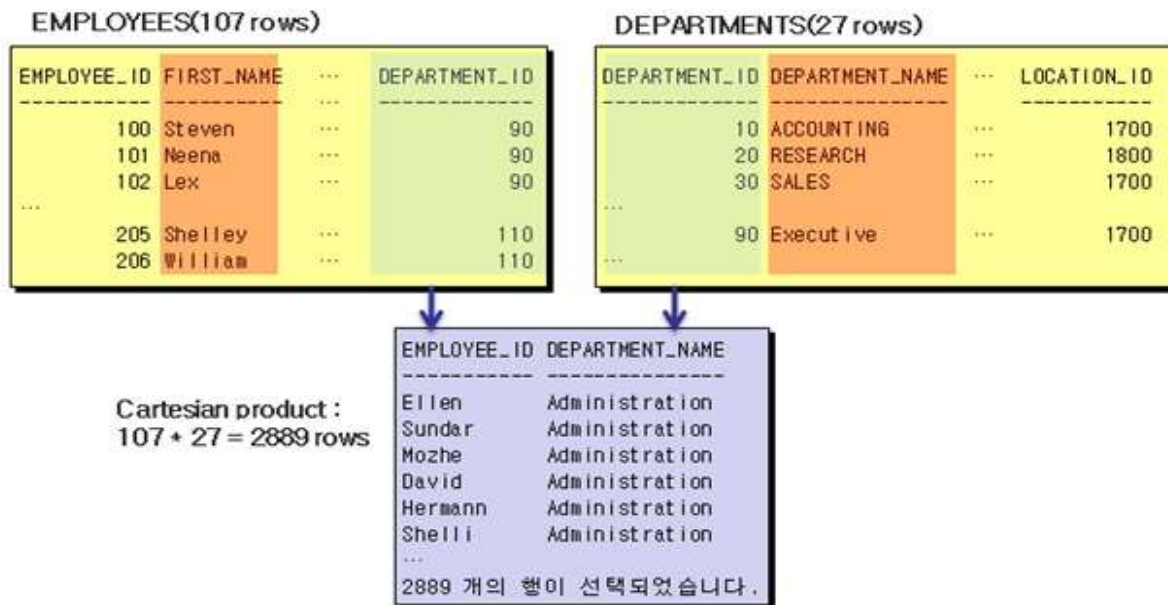
	EMPLOYEE_ID	FIRST_NAME	HIRE_DATE	START_DATE	END_DATE	JOB_ID	DEPARTMENT_ID
1	101	Neena	05/09/21	97/09/21	01/10/27	AC_ACCOUNT	110
2	101	Neena	05/09/21	01/10/28	05/03/15	AC_MGR	110
3	102	Lex	01/01/13	01/01/13	06/07/24	IT_PROG	60
4	114	Den	02/12/07	06/03/24	07/12/31	ST_CLERK	50
5	122	Payam	03/05/01	07/01/01	07/12/31	ST_CLERK	50
6	176	Jonathon	06/03/24	06/03/24	06/12/31	SA_REP	80
7	176	Jonathon	06/03/24	07/01/01	07/12/31	SA_MAN	80
8	200	Jennifer	03/09/17	02/07/01	06/12/31	AC_ACCOUNT	90
9	200	Jennifer	03/09/17	95/09/17	01/06/17	AD_ASST	90
10	201	Michael	04/02/17	04/02/17	07/12/19	MK_REP	20

크로스 조인은 두 개의 테이블에 대한 Cartesian Product와 같은 결과를 출력합니다. 의도적으로 데이터를 복제하기 위해 카티션 프로젝트를 사용한 것이 아니라면, 크로스 조인은 사용하지 않는 것이 바람직합니다.



```
SELECT    table1.column1, table2.column2
FROM      table1
CROSS JOIN table2
```

- Cartesian product는 다음의 경우에 발생합니다.
 - 조인 조건이 생략된 경우
 - 조인 조건이 잘못된 경우
 - 첫 번째 테이블의 모든 행이 두 번째 테이블의 모든 행과 조인되는 경우



문제 1.

-EMPLOYEES 테이블과, DEPARTMENTS 테이블은 DEPARTMENT_ID로 연결되어 있습니다.

-EMPLOYEES, DEPARTMENTS 테이블을 엘리어스를 이용해서
각각 INNER , LEFT OUTER, RIGHT OUTER, FULL OUTER 조인 하세요. (달라지는 행의 개수 확인)

문제 2.

-EMPLOYEES, DEPARTMENTS 테이블을 INNER JOIN하세요

조건)employee_id가 200인 사람의 이름, department_id를 출력하세요

조건)이름 컬럼은 first_name과 last_name을 합쳐서 출력합니다

문제 3.

-EMPLOYEES, JOBS테이블을 INNER JOIN하세요

조건) 모든 사원의 이름과 직무아이디, 직무 타이틀을 출력하고, 이름 기준으로 오름차순 정렬
HINT) 어떤 컬럼으로 서로 연결되 있는지 확인

문제 4.

--JOBS테이블과 JOB_HISTORY테이블을 LEFT_OUTER JOIN 하세요.

문제 5.

--Steven King의 부서명을 출력하세요.

문제 6.

--EMPLOYEES 테이블과 DEPARTMENTS 테이블을 Cartesian Product(Cross join)처리하세요

문제 7.

--EMPLOYEES 테이블과 DEPARTMENTS 테이블의 부서번호를 조인하고 SA_MAN 직원만의 직원번호, 이름, 급여, 부서명, 근무지를 출력하세요. (Alias를 사용)

문제 8.

-- employees, jobs 테이블을 조인 지정하고 job_title이 'Stock Manager', 'Stock Clerk'인 직원 정보만 출력하세요.

문제 9.

-- departments 테이블에서 직원이 없는 부서를 찾아 출력하세요. LEFT OUTER JOIN 사용

문제 10.

-join을 이용해서 사원의 이름과 그 사원의 매니저 이름을 출력하세요
힌트) EMPLOYEES 테이블과 EMPLOYEES 테이블을 조인하세요.

문제 11.

--6. EMPLOYEES 테이블에서 left join하여 관리자(매니저)와, 매니저의 이름, 매니저의 급여 까지 출력하세요
--매니저 아이디가 없는 사람은 배제하고 급여는 역순으로 출력하세요