chapter03

그룹함수
JOIN

1. **그룹함수**
2. GROUP BY절
3. HAVING절
4. CASE ~ END 문
5. JOIN

01 그룹함수

■ 그룹 함수의 특징

- 여러행으로부터 하나의 결과값을 반환
- 집계함수, 그룹함수, 복수행 함수

- 종류
  - avg() : 평균값을 반환합니다.
  - count() : 총 건 수를 반환합니다.
    - ✓ count(*)
    - ✓ count(컬럼명)
  - max() : 최대값을 반환합니다.
  - min() : 최소값을 반환합니다.
  - sum() : 합계를 반환합니다.

01 그룹함수

■ 그룹 함수의 특징

```
select salary, first_Name
from employees;
```

FIRST_NAME	SALARY
Steven	24000
Neena	17000
Lex	17000
Alexander	9000
Bruce	6000
David	4800
Valli	4800
Diana	4200
Nancy	12008
Daniel	9000
John	8200
Ismael	7700
Jose Manuel	7800
Luis	6900
Den	11000
Alexander	3100
Shelli	2900

- 그룹함수의 결과는 한 row만 남게 된다.
- department_id은 하나의 row에 표현될 수 없다.
- 부서별로 연봉 평균이 필요한 경우 **Group by절 사용**

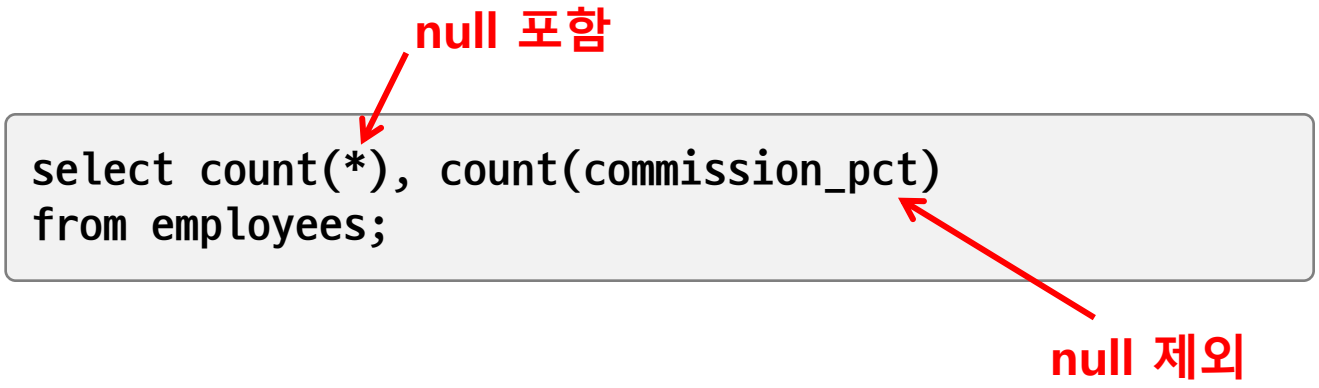
```
select avg(salary), ~~department_id~~
from employees;
```

?

AVG(SALARY)
6461.8317757009345794...

■ 그룹함수 - count()

- 함수에 입력되는 데이터의 총 건수를 구하는 함수




```
select count(*)
from employees
where salary > 16000;
```

■ 그룹함수 - sum()

- 입력된 데이터들의 합계 값을 구하는 함수

```
select count(*), sum(salary)
from employees;
```

■ 그룹함수 - avg()

- 입력된 값들의 평균값을 구하는 함수
- 주의: null 값이 있는 경우 빼고 계산함 – nvl 함수와 같이 사용

```
select count(*), sum(salary), avg(salary)
from employees;
```

name	point
홍길동	70
일지매	null → 0
유관순	50

- 120 / 3 = 40
- 120 / 2 = 60 ✓

```
select count(*), sum(salary), avg(nvl(salary,0))
from employees;
```

커미션 비율이 있는 사람들의 수, 커미션의 총합, 커미션 비율의 평균

■ 그룹함수 - max() / min()

- 입력된 값들중 가장 큰값/작은값 을 구하는 함수
- 여러건의 데이터를 순서대로 정렬 후 값을 구하기때문에 데이터가 많을 때는 느리다 (주의해서 사용)

```
select count(*), max(salary), min(salary)
from employees;
```

chapter03

그룹함수
JOIN

1. 그룹함수
2. **GROUP BY절**
3. HAVING절
4. CASE ~ END 문
5. JOIN

■ GROUP BY 절

```
select department_id, salary
from employees
order by department_id asc;
```

DEPARTMENT_ID	SALARY
10	4400
20	13000
20	6000
30	11000
30	3100
30	2900
30	2800
30	2600
30	2500
40	6500
50	8000
50	8200
50	7900
50	6500
50	5800
50	3200
50	2700

```
select department_id, avg(salary)
from employees
group by department_id
order by department_id asc;
```

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
30	4150
40	6500
50	3475.55555...

02 GROUP BY 절

■ GROUP BY 절 – 자주하는 오류

Group by에 참여한 컬럼이나 그룹함수만 올 수 있다.

```
select department_id, count(*), sum(salary)
from employees
group by department_id;
```

O

```
select department_id, job_id, count(*), sum(salary)
from employees
group by department_id;
```

X

```
select department_id, job_id, count(*), sum(salary)
from employees
group by department_id, job_id;
```

O

02 GROUP BY 절

■ GROUP BY 절 예제

[예제]

▪연봉(salary)의 합계가 20000 이상인 부서의 부서 번호와 , 인원수, 급여합계를 출력하세요

```
select department_id, count(*), sum(salary)
from employees
where sum(salary) > 20000
group by department_id;
```

■ GROUP BY 절 예제

[예제]

▪연봉(salary)의 합계가 20000 이상인 부서의 부서 번호와 , 인원수, 급여합계를 출력하세요

```
select department_id, count(*), sum(salary)
from employees
where sum(salary) > 20000
group by department_id;
```

where 절에는 그룹함수를 쓸 수 없다.

chapter03

그룹함수
JOIN

1. 그룹함수
2. GROUP BY절
3. **HAVING절**
4. CASE ~ END 문
5. JOIN

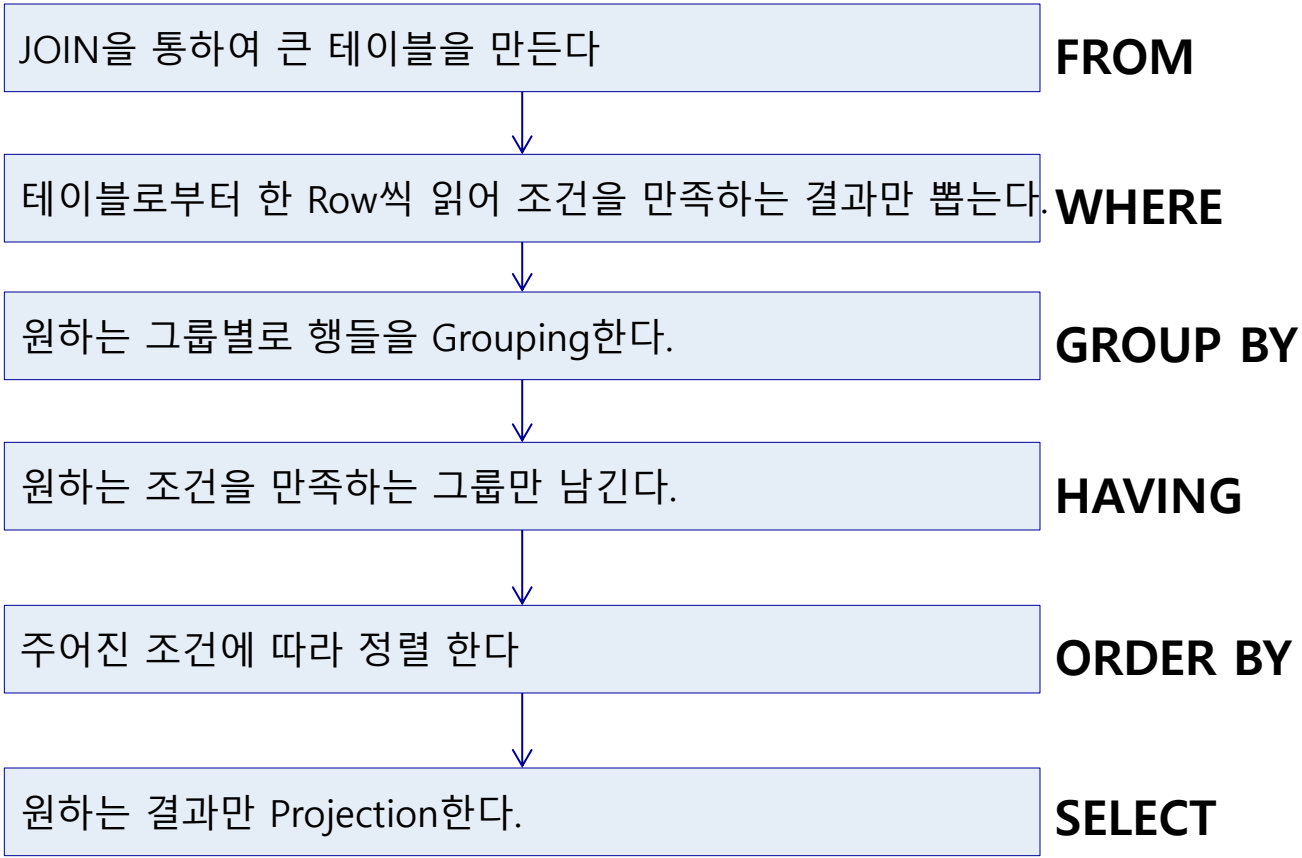
■ HAVING 절

```
select department_id, count(*), sum(salary)
from employees
group by department_id
having sum(salary) > 20000;
```

having 절에는
그룹함수 와
Group by에 참여한 컬럼만 사용할 수 있다.

```
select department_id, count(*), sum(salary)
from employees
group by department_id
having sum(salary) > 20000
and department_id = 100;
```

03 GROUP BY~HAVING 절



chapter03

그룹함수
JOIN

1. 그룹함수
2. GROUP BY절
3. HAVING절
4. **CASE ~ END 문**
5. JOIN

04 CASE ~ END 문 / DECODE() 함수

■ CASE ~ END 문

- if ~else if~else 문과 유사

```
CASE   WHEN 조건 THEN   출력1
      [WHEN 조건 THEN   출력2]  ← 필요시 조건 추가
       ELSE   출력3
END   "컬럼Alias"
```

```
SELECT employee_id,
       salary,
       CASE WHEN job_id = 'AC_ACCOUNT' THEN salary + salary * 0.1
            WHEN job_id = 'AC_MGR' THEN salary + salary *0.2
            ELSE salary
       END job_id
FROM employees;
```

04 CASE ~ END 문 / DECODE() 함수

[예제]

▪직원의 이름, 부서, 팀을 출력하세요
팀은 부서코드로 결정하며 부서코드가 10~50 이면 'A-TEAM'
60~100이면 'B-TEAM' 110~150이면 'C-TEAM' 나머지는 '팀없음' 으로 출력하세요

chapter03

그룹함수
JOIN

1. 그룹함수
2. GROUP BY절
3. HAVING절
4. CASE ~ END 문
5. **JOIN**

■ 직원의 이름과 직원이 속한 부서명을 함께 보고 싶다면

FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME	DEPARTMENT_ID_1
Jennifer	10	Administration	10
Pat	20	Marketing	20
Michael	20	Marketing	20
Sigal	30	Purchasing	30
Karen	30	Purchasing	30
Shelli	30	Purchasing	30

employees 직원테이블

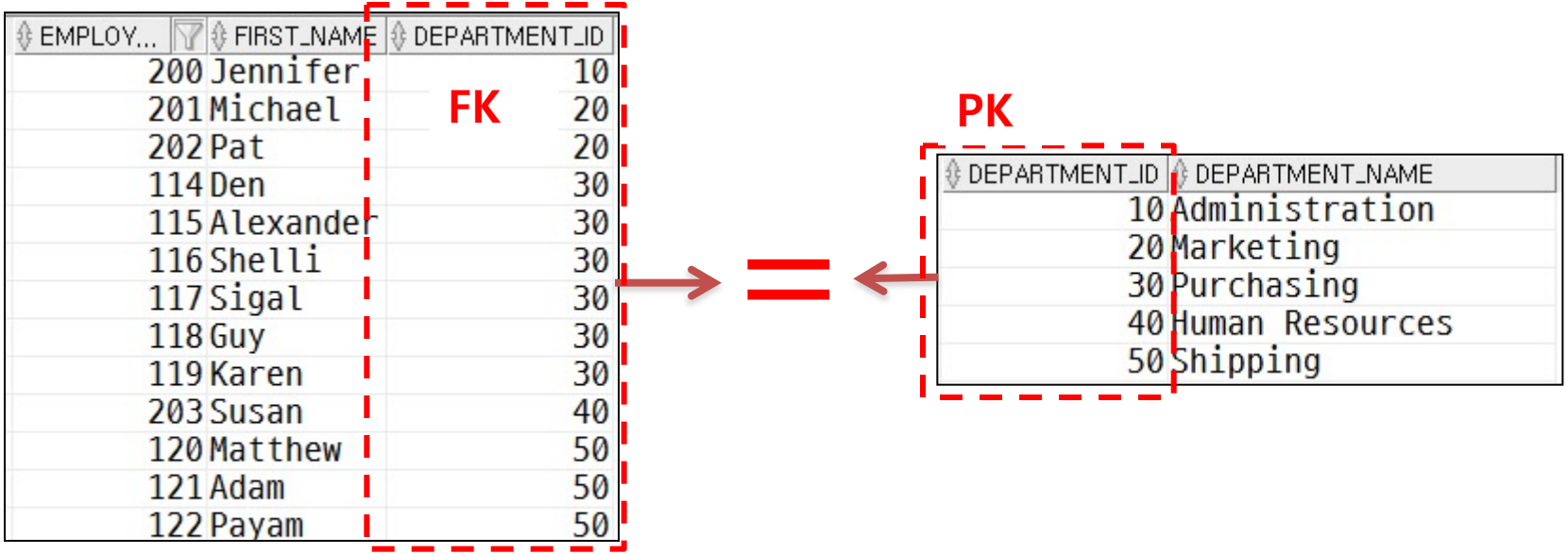
EMPLOY...	FIRST_NAME	DEPARTMENT_ID
200	Jennifer	10
201	Michael	20
202	Pat	20
114	Den	30
115	Alexander	30
116	Shelli	30
117	Sigal	30
118	Guy	30
119	Karen	30
203	Susan	40
120	Matthew	50

department 부서테이블

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
30	Purchasing
40	Human Resources
50	Shipping

■ 둘 이상의 테이블을 합쳐서 하나의 큰 테이블로 만드는 방법

- 관계형 모델에서는 데이터의 일관성이나 효율을 위하여 데이터의 중복을 최소화 (정규화)
- Foreign Key를 이용하여 참조
- 정규화된 테이블로부터 결합된 형태의 정보를 추출할 필요가 있음
- ex) **직원의 이름과 직원이 속한 부서명을 함께 보고 싶다면??**



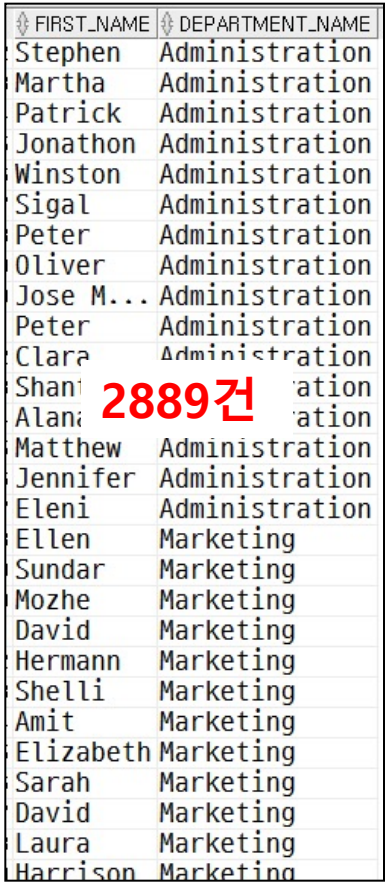
■ 두 테이블에서 그냥 결과를 선택하면?

```
select first_name, department_name
from employees, departments;
```

- 결과: 두 테이블의 행들의 가능한 모든 쌍이 추출됨
- 일반적으로 사용자가 원하는 결과가 아님.
- 107*27=2889

■ 카티젼 프로덕트(Cartesian Product)

- 올바른 Join조건을 WHERE 절에 부여 해야 함.



■ EQUI Join

```
select first_name, em.department_id,
       department_name, de.department_id
from   employees em, departments de
where  em.department_id = de.department_id;
```

FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME	DEPARTMENT_ID_1
Jennifer	10	Administration	10
Pat	20	Marketing	20
Michael	20	Marketing	20
Sigal	30	Purchasing	30
Karen	30	Purchasing	30
Shelli	30	Purchasing	30
Den	30	Purchasing	30
Alexander	30	Purchasing	30
Guy	30	Purchasing	30
Susan	40	Human Resources	40
Kevin	50	Shipping	50
Jean	50	Shipping	50

106건임(107건X)
양쪽다 만족하는 경우만 조인됨
→null은 조인안됨(제외됨)

■ EQUI Join

[예제]

▪직원의 이름, 직급명칭을 출력하세요
Join 할 테이블 (Employees, Jobs)

■ 설명

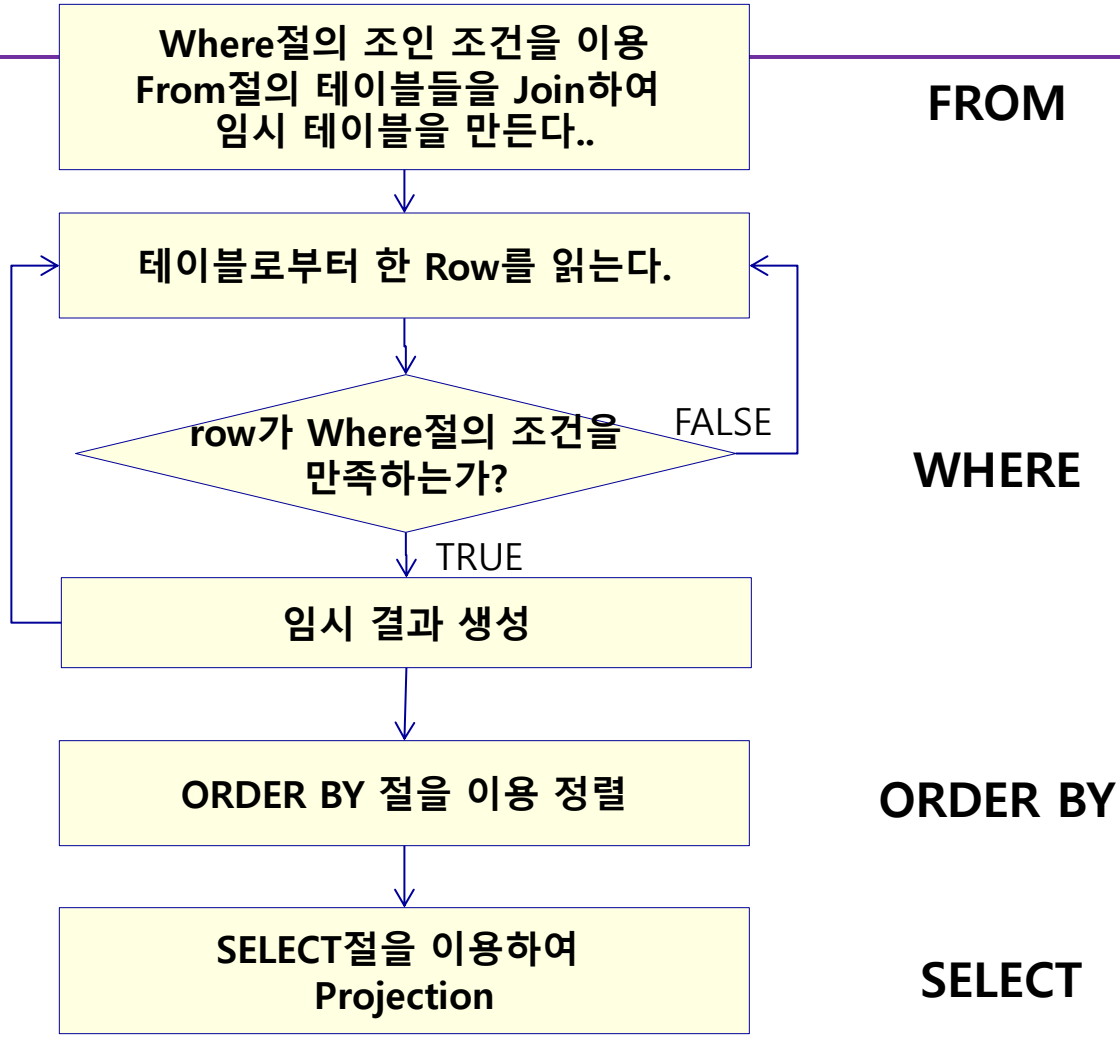
- FROM 절에 필요로 하는 테이블을 모두 적는다.
- 컬럼 이름의 모호성을 피하기 위해(어느 테이블 에 속하는지 알 수 없음)이 있을 수 있으므로 Table 이름에 Alias 사용 (테이블 이름으로 직접 지칭 가능)
- 적절한 Join 조건을 Where 절에 부여 (일반적으로 테이블 개수 -1 개의 조인 조건이 필요)
- 일반적으로 PK와 FK간의 = 조건이 붙는 경우가 많음

```
select first_name, em.department_id,
       department_name, de.department_id
from   employees em, departments de
where  em.department_id = de.department_id;
```

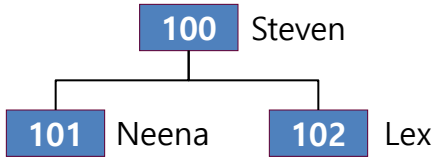
[예제]

▪모든 직원이름, 부서이름, 업무명 을 출력하세요

■ JOIN 처리방법


Where절의 조인 조건을 이용
From절의 테이블들을 Join하여
임시 테이블을 만든다..
FROM
테이블로부터 한 Row를 읽는다.
테이블의 모든 Row를
처리할 때까지 반복
row가 Where절의 조건을
만족하는가?
FALSE
WHERE
TRUE
임시 결과 생성
ORDER BY 절을 이용 정렬
ORDER BY
SELECT절을 이용하여
Projection
SELECT

사원 이름 과 그 사원의 매니저이름 조회



	EMPLOYEE_ID	FIRST_NAME	MANAGER_ID
1	100	Steven	[NULL]
2	101	Neena	100
3	102	Lex	100
4	103	Alexander	102
5	104	Bruce	103
6	105	David	103
7	106	Valli	103
8	107	Diana	103
9	108	Nancy	101
10	109	Daniel	108

Emp.manager_id

	EMPLOYEE_ID	FIRST_NAME	MANAGER_ID
1	100	Steven	[NULL]
2	101	Neena	100
3	102	Lex	100
4	103	Alexander	102
5	104	Bruce	103
6	105	David	103
7	106	Valli	103
8	107	Diana	103
9	108	Nancy	101
10	109	Daniel	108

Mgr.employee_id

```
Select emp.first_name, mgr.first_name
From employees emp, employees mgr
Where emp.manager_id = mgr.employee_id
```

■ OUTER Join

- Join 조건을 만족하지 않는 컬럼이 없는 경우 Null을 포함하여 결과를 생성
- 모든 행이 결과 테이블에 참여
- NULL이 올 수 있는 쪽 조건에 (+)를 붙인다.

■ 종류

- Left Outer Join: 왼쪽의 모든 튜플은 결과 테이블에 나타남
- Right Outer Join: 오른쪽의 모든 튜플은 결과 테이블에 나타남
- Full Outer Join: 양쪽 모두 결과 테이블에 참여

■ left outer join

employees

106개

1개

EMPLOYEE_ID	FIRST_NAME	DEPARTMENT_ID
100	Steven	90
101	Neena	90
102	Lex	90
108	Nancy	100
109	Daniel	100
110	John	100
111	Ismael	100
112	Jose M...	100
113	Luis	100
205	Shelley	110
206	William	110
178	Kimberely	(null)

departments

	DEPARTMENT_ID	DEPARTMENT_NAME
1	10	Administration
2	20	Marketing
3	30	Purchasing
4	40	Human Resources
5	50	Shipping
6	60	IT
7	70	Public Relations
8	80	Sales
9	90	Executive
10	100	Finance
11	110	Accounting
12	120	Treasury

107개

FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME	DEPARTMENT_ID_1
Jose M...	100	Finance	100
Ismael	100	Finance	100
John	100	Finance	100
Daniel	100	Finance	100
Nancy	100	Finance	100
William	110	Accounting	110
Shelley	110	Accounting	110
Kimberely	(null)	(null)	(null)

■ left outer join

- 왼쪽 테이블의 모든 row를 결과 테이블에 나타냄

```
select e.department_id, e.first_name, d.department_name
  from employees e left outer join departments d
    on e.department_id = d.department_id ;
```

```
select e.department_id, e.first_name, d.department_name
  from employees e, departments d
 where e.department_id = d.department_id(+) ;
```

■ right outer join

employees

EMPLOYEE_ID	FIRST_NAME	DEPARTMENT_ID
100	Steven	90
101	Neena	90
102	Lex	90
108	Nancy	100
109	Daniel	100
110	John	100
111	Ismael	100
112	Jose M...	100
113	Luis	100
205	Shelley	110
206	William	110
178	Kimberely	(null)

106개

1개

departments

	DEPARTMENT_ID	DEPARTMENT_NAME
1	10	Administration
2	20	Marketing
3	30	Purchasing
4	40	Human Resources
5	50	Shipping
6	60	IT
7	70	Public Relations
8	80	Sales
9	90	Executive
10	100	Finance
11	110	Accounting
12	120	Treasury

11개 (사용o)

16개 (사용x)

103	100	Ismael	Finance
104	100	Nancy	Finance
105	110	William	Accounting
106	110	Shelley	Accounting
107	(null)	(null)	Treasury
108	(null)	(null)	Corporate Tax
109	(null)	(null)	Control And Cr...
110	(null)	(null)	Shareholder Se...
111	(null)	(null)	Benefits
112	(null)	(null)	Manufacturing
113	(null)	(null)	Construction
114	(null)	(null)	Contracting
115	(null)	(null)	Operations
116	(null)	(null)	IT Support
117	(null)	(null)	NOC
118	(null)	(null)	IT Helpdesk
119	(null)	(null)	Government Sales
120	(null)	(null)	Retail Sales
121	(null)	(null)	Recruiting
122	(null)	(null)	Payroll

106개

16개

122개

■ Right outer join

- 오른쪽 테이블의 모든 row를 결과 테이블에 나타냄

```
select e.department_id, e.first_name, d.department_name
  from employees e right outer join departments d
    on e.department_id = d.department_id ;
```

```
select e.department_id, e.first_name, d.department_name
  from employees e, departments d
 where e.department_id(+) = d.department_id ;
```

■ right outer join → left outer join

departments

	DEPARTMENT_ID	DEPARTMENT_NAME
1	10	Administration
2	20	Marketing
3	30	Purchasing
4	40	Human Resources
5	50	Shipping
6	60	IT
7	70	Public Relations
8	80	Sales
9	90	Executive
10	100	Finance
11	110	Accounting
12	120	Treasury

11개 (사용o)

16개 (사용x)

employees

EMPLOYEE_ID	FIRST_NAME	DEPARTMENT_ID
100	Steven	90
101	Neena	90
102	Lex	90
103	Nancy	100
104	Daniel	100
105	John	100
106	Ismael	100
107	Jose M...	100
108	Luis	100
109	Shelley	110
110	William	110
111	Kimberely	(null)

106개

1개

103	100	Ismael	Finance
104	100	Nancy	Finance
105	110	William	Accounting
106	110	Shelley	Accounting
107	(null)	(null)	Treasury
108	(null)	(null)	Corporate Tax
109	(null)	(null)	Control And Cr...
110	(null)	(null)	Shareholder Se...
111	(null)	(null)	Benefits
112	(null)	(null)	Manufacturing
113	(null)	(null)	Construction
114	(null)	(null)	Contracting
115	(null)	(null)	Operations
116	(null)	(null)	IT Support
117	(null)	(null)	NOC
118	(null)	(null)	IT Helpdesk
119	(null)	(null)	Government Sales
120	(null)	(null)	Retail Sales
121	(null)	(null)	Recruiting
122	(null)	(null)	Payroll

106개

16개

122개

■ full outer join

```
select e.department_id, e.first_name, d.department_name
  from employees e full outer join departments d
    on e.department_id = d.department_id ;
```

76	80	Alyssa	Sales
77	80	Jonathon	Sales
78	80	Jack	Sales
79	(null)	Kimberely	(null)
80	80	Charles	Sales
81	50	Winston	Shipping
82	50	Jean	Shipping
83	50	Martha	Shipping
84	50	Girard	Shipping
85	50	Nandita	Shipping
86	50	Alexis	Shipping
87	50	Julia	Shipping

105	70	Hermann	Public Relations
106	110	Shelley	Accounting
107	110	William	Accounting
108	(null)	(null)	NOC
109	(null)	(null)	Manufacturing
110	(null)	(null)	Government Sales
111	(null)	(null)	IT Support
112	(null)	(null)	Benefits
113	(null)	(null)	Shareholder Se...
114	(null)	(null)	Retail Sales
115	(null)	(null)	Control And Cr...
116	(null)	(null)	Recruiting
117	(null)	(null)	Operations
118	(null)	(null)	Treasury
119	(null)	(null)	Payroll
120	(null)	(null)	Corporate Tax
121	(null)	(null)	Construction
122	(null)	(null)	Contracting
123	(null)	(null)	IT Helpdesk

■ OUTER Join

FK

EMPLOYEE_ID	FIRST_NAME	DEPARTMENT_ID
100	Steven	90
101	Neena	90
102	Lex	90
108	Nancy	100
109	Daniel	100
110	John	100
111	Ismael	100
112	Jose M...	100
113	Luis	100
205	Shelley	110
206	William	110
178	Kimberely	(null)

PK

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
30	Purchasing
40	Human Resources
50	Shipping
60	IT
70	Public Relations
80	Sales
90	Executive
100	Finance
110	Accounting

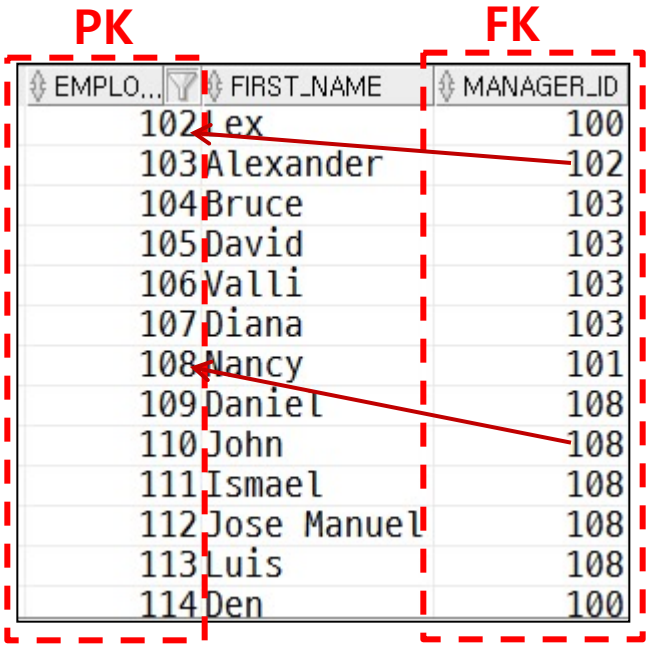
```
...
where em.department_id = de.department_id(+);
```

NULL이 올 수 있는 쪽 조건에 (+)를 붙인다.

FIRST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME	DEPARTMENT_ID_1
William	110	Accounting	110
Shelley	110	Accounting	110
Kimberely	(null)	(null)	(null)

■ Self Join

- 자기자신과 Join
- Alias를 사용할 수 밖에 없음



EMPLOYEE_ID	FIRST_NAME	MANAGER_ID	MANAGER
102	Lex	100	Steven
103	Alexander	102	Lex
104	Bruce	103	Alexander
105	David	103	Alexander
106	Valli	103	Alexander
107	Diana	103	Alexander
108	Nancy	101	Neena
109	Daniel	108	Nancy
110	John	108	Nancy
111	Ismael	108	Nancy

■ Self Join

employees emp

EMPLO...	FIRST_NAME	MANAGER_ID
102	Lex	100
103	Alexander	102
104	Bruce	103
105	David	103
106	Valli	103
107	Diana	103
108	Nancy	101
109	Daniel	108
110	John	108
111	Ismael	108

FK

=

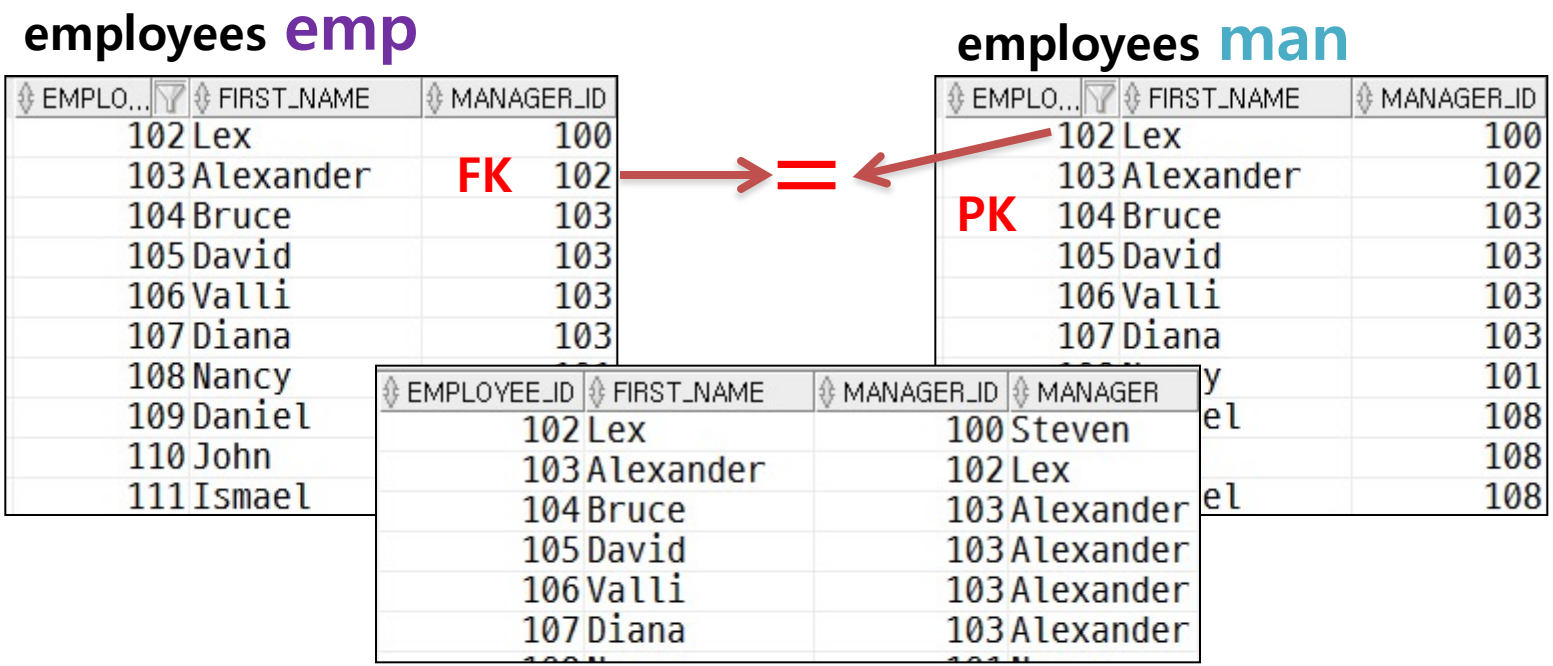
PK

employees man

EMPLO...	FIRST_NAME	MANAGER_ID
102	Lex	100
103	Alexander	102
104	Bruce	103
105	David	103
106	Valli	103
107	Diana	103
108	Nancy	101
109	Daniel	108
110	John	108
111	Ismael	108

EMPLOYEE_ID	FIRST_NAME	MANAGER_ID	MANAGER
102	Lex	100	Steven
103	Alexander	102	Lex
104	Bruce	103	Alexander
105	David	103	Alexander
106	Valli	103	Alexander
107	Diana	103	Alexander

```
select emp.employee_id, emp.first_name,
       emp.manager_id, man.first_name manager
from employees emp, employees man
where emp.manager_id = man.employee_id
```




```
select emp.employee_id, emp.first_name,
       emp.manager_id, man.first_name manager
from employees emp, employees man
where emp.manager_id = man.employee_id
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	SALA...
132	TJ	Olson	TJOLSON	650.124.8234	2100
128	Steven	Markle	SMARKLE	650.124.1434	2200
136	Hazel	Philtanker	HPHILTAN	650.127.1634	2200
127	James	Landry	JLANDRY	650.124.1334	2400
135	Ki	Gee	KGEE	650.127.1734	2400
119	Karen	Colmenares	KCOLMENA	515.127.4566	2500
131	James	Marlow	JAMRLOW	650.124.7234	2500
140	Joshua	Patel	JPATEL	650.121.1834	2500
144	Peter	Vargas	PVARGAS	650.121.2004	2500
182	Martha	Sullivan	MSULLIVA	650.507.9878	2500
191	Randall	Perkins	RPERKINS	650.505.4876	2500
118	Guy	Himuro	GHIMURO	515.127.4565	2600
143	Randall	Matos	RMATOS	650.121.2874	2600
198	Donald	OConnell	DOCONNEL	650.507.9833	2600
199	Douglas	Grant	DGRANT	650.507.9844	2600
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	2700
139	John	Seo	JSEO	650.121.2019	2700
117	Sigal	Tobias	STOBIAS	515.127.4564	2800
130	Mozhe	Atkinson	MATKINSO	650.124.6234	2800
183	Girard	Geoni	GGEONI	650.507.9879	2800
195	Vance	Jones	VJONES	650.501.4876	2800
116	Shelli	Baida	SBAIDA	515.127.4563	2900
134	Michael	Rogers	MROGERS	650.127.1834	2900
190	Timothy	Gates	TGATES	650.505.3876	2900
187	Anthony	Cabrio	ACABRIO	650.509.4876	3000
197	Kevin	Feeney	KFEENEY	650.507.9822	3000
115	Alexander	Khoo	AKHOO	515.127.4562	3100
142	Curtis	Davies	CDAVIES	650.121.2994	3100
181	Jean	Fleaur	JFLEAUR	650.507.9877	3100
196	Alana	Wals			
125	Julia	Naye			
138	Stephen	Stil			
180	Winston	Taylor	WTAYLOR	650.507.9876	3200
194	Samuel	McCain	SMCCAIN	650.501.3876	3200

34건

LOCATION_ID	STREET_ADDRESS	POSTAL_COD
1000	1297 Via Cola di Rie	00989
1100	93091 Calle della ...	10934
1200	2017 Shinjuku-ku	1689
1300	9450 Kamiya-cho	6823
1400	2014 Jabberwocky Rd	26192
1500	2011 Interiors Blvd	99236
1600	2007 Zagora St	50090
1700	2004 Charade Rd	98199
1800	147 Spadina Ave	M5V 2L7
1900	6092 Boxwood St	YSW 9T2
2000	40-5-12 Laogianggen	190518
2100	1298 Vileparle (E)	490231
2200	12-98 Victoria Street	2901
2300	198 Clementi North	540198
2400	8204 Arthur St	(null)
2500	Magdalen Centre, T...	OX9 9ZB
2600	9702 Chester Road	096298502
2700	Schwanthalerstr. 7031	80925
2800	Rua Frei Caneca 1360	01307-002
2900	20 Rue des Corps-S...	1730
3000	Murtenstrasse 921	3095
3100	Pieter Breughelstr...	3029SK
3200	Mariano Escobedo 9991	11932

employees.salary = **locations.location_id**