

# Day29; 20220809

강의 번호	
유형	@2022년 8월 9일
자료	
복습	<input type="checkbox"/>
속성	
작성일자	@2022년 8월 9일 오전 9:29

## Oracle

그룹함수 - 단일행 함수와는 달리 여러 행에 대해 함수가 적용되어 하나의 결과를 나타내는 함수이다. 예를 들면 학생들의 영어점수들의 합계, 수학 점수들의 평균... 열에 대해 같은 값끼리 묶어 행들의 집합에 대해 그룹 연산이 필요할 때는 GROUP BY 절을 사용하여 처리하고 또 묶은 그룹에 다시 조건이 필요할 때는 HAVING 절을 이용한다.

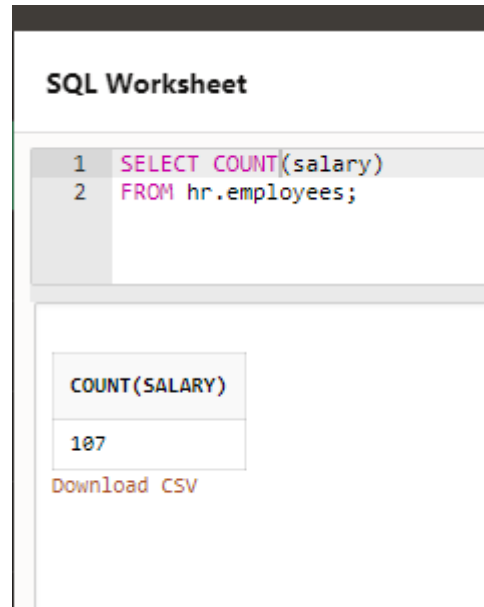
SELECT 그룹 함수(열이름)
FROM 테이블 이름
[WHERE 조건식]
[ORDER BY 열이름];

COUNT	개수	null 의 값도 개수로 셈
SUM	합계	나머지 아래 함수들은 전부 null값은 제외하고 수행한다.
AVG	평균	
MAX	최대값	
MIN	최소값	
STDDEV	표준편차	
VARIANCE	분산	

### Quiz 1

employees 테이블에서 salary 의 행의 개수가 몇개인지 세어서 출력하세요

- 주의 - count함수는 null값도 행으로 셀수 있다는 점을 유의하자.



```
SELECT COUNT(salary)
FROM hr.employees;
```

### Quiz 2

employees 테이블에서 salary 의 합계와 평균을 구하세요 또한 avg함수를 사용하지 말고 salary 의 평균을 구하세요

합계      평균      계산평균

- AVG 함수는 null값을 제외하고 평균을 구하는데 null 값을 포함하여 평균을 계산해야 한다면 뒤에서 배울 NVL 함수를 사용하여 구할 수 있다.

### SQL Worksheet

Clear

Find

```
1 SELECT SUM(salary) 합계, AVG(salary) 평균, SUM(salary)/ COUNT(salary) 계산평균
2 FROM hr.employees;
```

합계	평균	계산평균
691416	6461.831775700934579439252336448598130841	6461.831775700934579439252336448598130841

Download CSV

```
SELECT SUM(salary) 합계, AVG(salary) 평균, SUM(salary)/ COUNT(salary) 계산평균
FROM hr.employees;
```

### Quiz 3

employees 테이블에서 salary 의 최대값과 최소값 first\_name의 최대값과 최소값을 출력해보세요

최대 월급      최소 월급      이름 최대      이름 최소

### SQL Worksheet

Clear

Find

Actions

Save

Run

```
1 SELECT MAX(salary) "최대 월급", MIN(salary) "최소 월급", MAX(first_name) "이름 최대", MIN(first_name) "이름 최소"
2 FROM hr.employees;
```

최대 월급	최소 월급	이름 최대	이름 최소
24000	2100	Winston	Adam

Download CSV

## GROUP BY

특정 열의 데이터 값들을 그룹으로 묶어 그룹함수를 적용한다.

SELECT 절에 열이름과 그룹 함수를 함께 기술할 때는 GROUP BY 절을 사용하게 된다.  
그룹화는 열이름이 기술된 순서대로 수행된다.

```
SELECT 기준열, 그룹함수(열이름)
FROM 테이블이름
[ WHERE 조건식]
GROUP BY 열이름
ORDER BY 열이름 [ASC / DESC];
```

테이블에 접근하여

기술된 열이름을 기준으로 그룹화하여

선택 열들을 나열하고 지정된 열이름에 그룹함수를 적용하게 된다.

---

### Quiz 4

employees 테이블에서 employee\_id 가 10 이상인 직원들에 대해서 job\_id 별로 그룹화하여 job\_id 별 총 급여와 job\_id별 평균 급여를 구하고 job\_id 별 총 급여를 기준으로 내림차순 정렬하세요

직무	직무별 총급여	직무별 평균급여
----	---------	----------

```
1 SELECT job_id "직무", SUM(salary) "직무별 총급여", AVG(salary) "직무별 평균 급여"
2 FROM hr.employees
3 WHERE employee_id >= 10
4 GROUP BY job_id
5 ORDER BY job_id DESC;
```

직무	직무별 총급여	직무별 평균 급여
ST_MAN	36400	7280
ST_CLERK	55700	2785
SH_CLERK	64300	3215
SA_REP	250500	8350
SA_MAN	61000	12200
PU_MAN	11000	11000
PU_CLERK	13900	2780
PR_REP	10000	10000
MK_REP	6000	6000
MK_MAN	13000	13000
IT_PROG	28800	5760
HR_REP	6500	6500
FI_MGR	12008	12008
FI_ACCOUNT	39600	7920
AD_VP	34000	17000
AD_PRES	24000	24000
AD_ASST	4400	4400
AC_MGR	12008	12008
AC_ACCOUNT	8300	8300

[Download CSV](#)

19 rows selected.

```
SELECT job_id "직무", SUM(salary) "직무별 총급여", AVG(salary) "직무별 평균 급여"
FROM hr.employees
WHERE employee_id >= 10
GROUP BY job_id
ORDER BY job_id DESC;
```

## HAVING

그룹화된 값에 조건식을 적용할 때 사용된다. 즉, WHERE절에서는 그룹함수를 사용할 수는 없기 때문에 HAVING절을 사용하여 그룹함수의 결과값에 대해 조건식을 적용할 수 있다.

SELECT 기준열, 그룹함수(열이름)

FROM 테이블이름

[WHERE 조건식]

GROUP BY 열이름

[HAVING 조건식];

---

### Quiz 5

employees 테이블에서 employee\_id 가 10이상인 직원에 대해서 job\_id별로 그룹화하여 job\_id 별 총급여와 job\_id별 평균 급여를 구하되,,,,,

job\_id 별로 총 급여가 30000보다 큰 값 만 출력하는데 결과는 job\_id별 총 급여를 기준으로 내림차순 정렬하세요.

직무	직무별 총급여	직무별 평균급여
----	---------	----------

```
1 SELECT job_id "직무", SUM(salary) "직무별 총급여", AVG(salary) "직무별 평균 급여"
2 FROM hr.employees
3 WHERE employee_id >= 10
4 GROUP BY job_id
5 HAVING SUM(salary) > 30000
6 ORDER BY job_id DESC;
```

직무	직무별 총급여	직무별 평균 급여
ST_MAN	36400	7280
ST_CLERK	55700	2785
SH_CLERK	64300	3215
SA_REP	250500	8350
SA_MAN	61000	12200
FI_ACCOUNT	39600	7920
AD_VP	34000	17000

[Download CSV](#)

7 rows selected.

```
SELECT job_id "직무", SUM(salary) "직무별 총급여", AVG(salary) "직무별 평균 급여"
FROM hr.employees
WHERE employee_id >= 10
GROUP BY job_id
HAVING SUM(salary) > 30000
ORDER BY job_id DESC;
```

- WHERE 절에서는 집계함수를 사용 할 수 없다.
- HAVING 절은 집계함수를 가지고 조건비교를 할 때 사용한다.
- HAVING절은 GROUP BY절과 함께 사용이 된다.

## 조인 JOIN

관계형 데이터베이스의 의미는 테이블들이 관계를 맺고 조작되는 원리에서 유래되었다. 테이블들에는 각각의 성질에 맞는 데이터들이 저장되어 있고 그 테이블들은 특정한 규칙에 따라 상호관계를 맺는다. 데이터는 여러 테이블에 흩어져 저장되어 있어 사용자가 원하는대로 데이터를 사용하려면 특별한 방법이 필요하다. 이 때 사용되는 기법이 바로 조인이다.

조인은 한 개 이상의 테이블과 테이블을 서로 연결하여 사용하는 기법을 의미한다.

실무에서는 동등 조인, 외부 조인, 자체 조인 등을 사용하게 된다.

(이외에도 곱집합, 비동등조인 ... 다양한 조인들도 있다.)

```
SELECT 테이블명1.열이름1, 테이블명2.열이름2
FROM 테이블명1, 테이블명2
WHERE 테이블명1.열이름1 = 테이블명2.열이름2;
```

조인을 사용할 때 규칙

- SELECT 절에는 출력할 열이름들을 모두 기술한다.
- FROM절에는 접근하려는 테이블들을 기술한다.
- WHERE 절에는 조인 조건을 기술한다.
- 사용되는 테이블 이름들에는 별칭을 alias를 사용하면 편하다.

#### 1. 동등 조인 - 똑같은 데이터끼리 연결

동등 조인은 양 테이블에서 조인 조건이 일치하는 행들만 가져오는 가장 일반적으로 자주 사용되는 조인이다. = 등호 연산자를 사용하여 조건 값이 정확하게 일치할 때만 행을 가져오기 때문에 inner join 이라고도 부른다.



## SQL Worksheet

Clear

Find

Actions ▾

Save

Run

```

1 SELECT *
2 FROM hr.employees, hr.departments
3 WHERE employees.department_id = departments.department_id;

```

NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID	DEPARTMENT_ID	DEPARTMENT_NAME
John	JWHALEN	515.123.4444	17-SEP-03	AD_ASST	4400	-	101	10	10	Administration
Neena	MHARTSTE	515.123.5555	17-FEB-04	MK_MAN	13000	-	100	20	20	Marketing
Lex	PFAY	603.123.6666	17-AUG-05	MK_REP	6000	-	201	20	20	Marketing
Deena	DRAPHEAL	515.127.4561	07-DEC-02	PU_MAN	11000	-	100	30	30	Purchasing
Michael	AKHOO	515.127.4562	18-MAY-03	PU_CLERK	3100	-	114	30	30	Purchasing
Pat	SBAIDA	515.127.4563	24-DEC-05	PU_CLERK	2900	-	114	30	30	Purchasing
Scott	STOBIAS	515.127.4564	24-JUL-05	PU_CLERK	2800	-	114	30	30	Purchasing
Timothy	GHIMURO	515.127.4565	15-NOV-06	PU_CLERK	2600	-	114	30	30	Purchasing
Michael	KCOLMENA	515.127.4566	10-AUG-07	PU_CLERK	2500	-	114	30	30	Purchasing
David	SMAVRIS	515.123.7777	07-JUN-02	HR_REP	6500	-	101	40	40	Human Resources
Shelley	MNEISS	650.123.1234	18-JUL-04	ST_MAN	8000	-	100	50	50	Shipping
Stephanie	AFRIPP	650.123.2234	10-APR-05	ST_MAN	8200	-	100	50	50	Shipping
John	PKAUFLIN	650.123.3234	01-MAY-03	ST_MAN	7900	-	100	50	50	Shipping
Jan	SVOLLMAN	650.123.4234	10-OCT-05	ST_MAN	6500	-	100	50	50	Shipping
Joshua	KMOURGOS	650.123.5234	16-NOV-07	ST_MAN	5800	-	100	50	50	Shipping
John	JNAYER	650.124.1214	16-JUL-05	ST_CLERK	3200	-	120	50	50	Shipping
Lin	IMIKKILI	650.124.1224	28-SEP-06	ST_CLERK	2700	-	120	50	50	Shipping
Y	JLANDRY	650.124.1334	14-JAN-07	ST_CLERK	2400	-	120	50	50	Shipping
E	SMARKLE	650.124.1434	08-MAR-08	ST_CLERK	2200	-	120	50	50	Shipping
T	LBISSOT	650.124.5234	20-AUG-05	ST_CLERK	3300	-	121	50	50	Shipping
son	MATKINSO	650.124.6234	30-OCT-05	ST_CLERK	2800	-	121	50	50	Shipping
W	JAMLOW	650.124.7234	16-FEB-05	ST_CLERK	2500	-	121	50	50	Shipping
	TJOLSON	650.124.8234	10-APR-07	ST_CLERK	2100	-	121	50	50	Shipping

```

SELECT *
FROM hr.employees, hr.departments
WHERE employees.department_id = departments.department_id;

```

```

1 SELECT *
2 FROM hr.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

```

1 SELECT *
2 FROM hr.employees;

```

ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000	-	-	90
	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	-	100	90
	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	-	100	90
	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000	-	102	60

## Quiz 6

employees 테이블과 departments 테이블과 locations테이블을 조인하여 각 직원이 어느 부서에 속하는지와 부서의 소재지가 어디인지 조회하시오.

사원아이디    부서아이디    부서명    지역아이디    도시

```

1 SELECT *
2 FROM hr.employees;

```

DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
2003-03-15	AD_PRES	24000	-	-	90
2003-05-15	AD_VP	17000	-	100	90
2003-07-15	AD_VP	17000	-	100	90
2003-09-15	IT_PROG	9000	-	102	60

```

1 SELECT *
2 FROM hr.departments;

```

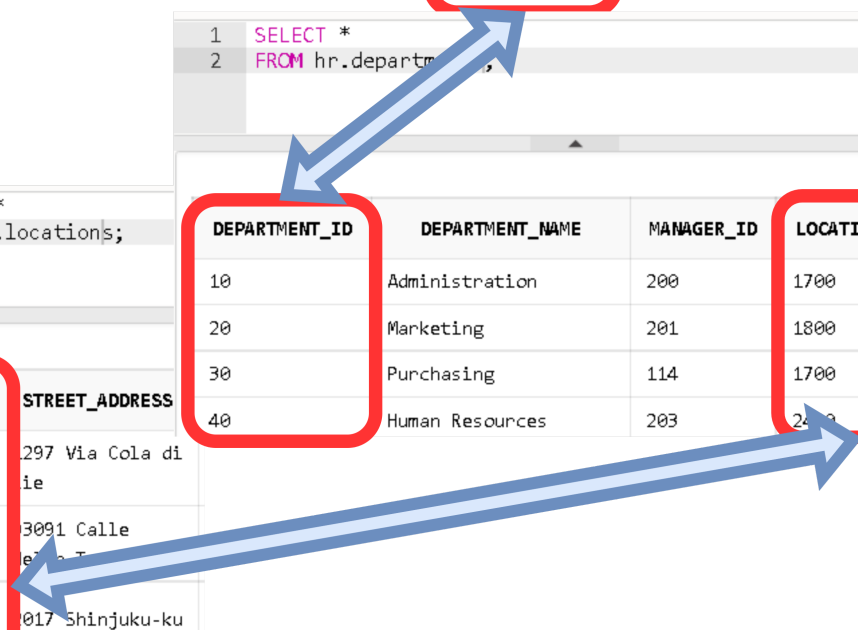
DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing	114	1700
40	Human Resources	203	2400

```

1 SELECT *
2 FROM hr.locations;

```

LOCATION_ID	STREET_ADDRESS
1000	297 Via Cola di Rienzo
1100	3091 Calle de la Reina
1200	1017 Shinjuku-ku
1300	450 Kamiya-cho
1400	1014 Jabberwocky Road



```

1 SELECT employees.employee_id "사원아이디", departments.department_id "부서아이디",
2 departments.department_name "부서명", departments.LOCATION_ID "지역아이디", locations.city "도시"
3 FROM hr.employees, hr.departments, hr.locations
4 WHERE employees.department_id = departments.department_id
5 AND departments.location_id = locations.location_id;

```

사원아이디	부서아이디	부서명	지역아이디	도시
100	90	Executive	1700	Seattle
101	90	Executive	1700	Seattle
102	90	Executive	1700	Seattle
103	60	IT	1400	Southlake
104	60	IT	1400	Southlake
105	60	IT	1400	Southlake
106	60	IT	1400	Southlake
107	60	IT	1400	Southlake
108	100	Finance	1700	Seattle
109	100	Finance	1700	Seattle
110	100	Finance	1700	Seattle

## 외부조인 - 모든 데이터를 연결

동등조인은 데이터 값이 정확히 일치하는 경우에만 결과를 출력한다. 데이터 값이 일치하지 않으면 결과가 조회되지 않는다.

```

1 SELECT count(*) "조인된 건수"
2 FROM hr.employees E, hr.departments D
3 WHERE E.department_id = D.department_id;

```

조인된 건수
106

```

1 SELECT count(*) "건수"
2 FROM hr.employees E;

```

건수
107

동등 조인의 결과는 106건이 출력되었는데 employees의 직원정보는 모두 107건이다. 1건의 차이가 나는 이유는 일치하지 않는 1건이 누락되었기 때문이다.

외부조인(outer join)은 조건을 만족하지 않는 행도 모두 출력하는 조인 기법이다.

- 외부 조인은 동등 조인 조건을 만족하지 못해 누락되는 행을 출력하기 위해 사용된다.
- 외부 조인은 (+) 기호를 사용한다. + 기호는 조인할 행이 없는 즉 데이터가 부족한 테이블의 열 이름 뒤에 기술한다.
- + 기호는 외부 조인 하려는 한쪽에만 기술할 수 있다. 테이블 양쪽에 모두 기술할 수는 없다.
- + 기호를 붙이면 데이터 값이 부족한 테이블에 NULL 값을 갖는 행이 생성되어 데이터 값이 충분한 테이블의 행들이 NULL행이 조인된다.

```

1 SELECT count(*) "조인된 건수"
2 FROM hr.employees E, hr.departments D
3 WHERE E.department_id = D.department_id(+);

```

조인된 건수

107

```

SELECT COUNT(*) "조인된 건수"
FROM hr.employees E, hr.departments D
WHERE E.department_id = D.department_id(+);

```

외부 조인은 동등 조인과 함께 실무에서 가장 많이 사용되는 조인이다.

- 양쪽 테이블 중 전부 출력하고 싶은 테이블 쪽을 앞에둔다.
- (+)는 다른쪽(뒤쪽) 테이블에 조인 조건에 붙인다.
- 

## 자체 조인 Self Join

employees 테이블의 직원 정보에는 manager\_id 열이 있다. 이 열은 그 직원의 담당 매니저의 정보를 담고 있는 열이다. 그러면 생각해보자. 어떤 직원의 담당 매니저를 알고 싶으면 그 테이블에 결국 다시 조인해야 한다. 이렇게 자기 자신의 테이블을 조인하는 것을 자체 조인 self join이라고 한다.

```

SELECT 열이름1, 열이름2, 열이름3
FROM 테이블1 이름 별명1, 테이블1 이름 별명2
WHERE 테이블1.열이름1 = 테이블1.열이름2;

```

한개의 테이블의 별명을 2개로 각각 다른 이름으로 설정한다.

employees 테이블을 자체조인하여 직원별 담당 매니저가 누구인지를 조회하세요.

사원아이디      사원이름      매니저아이디      매니저이름

### SQL Worksheet

Clear

```
1 SELECT a.employee_id "사원아이디", a.first_name || a.last_name "사원이름",  
2 a.manager_id "매니저 아이디", b.first_name || b.last_name "매니저이름"  
3 FROM hr.employees a, hr.employees b  
4 WHERE a.manager_id = b.employee_id;
```

사원아이디	사원이름	매니저 아이디	매니저이름
101	NeenaKochhar	100	StevenKing
102	LexDe Haan	100	StevenKing
114	DenRaphaely	100	StevenKing
120	MatthewWeiss	100	StevenKing
121	AdamFripp	100	StevenKing
122	PayamKaufling	100	StevenKing
123	ShantaVollman	100	StevenKing
124	KevinMourgos	100	StevenKing
145	JohnRussell	100	StevenKing
146	KarenPartners	100	StevenKing
...	...	...	...

```
SELECT a.employee_id "사원아이디", a.first_name || a.last_name "사원이름",  
a.manager_id "매니저 아이디", b.first_name || b.last_name "매니저이름"  
FROM hr.employees a, hr.employees b  
WHERE a.manager_id = b.employee_id;
```

## 서브 쿼리 - SELECT 문 안에 SELECT

서브 쿼리는 SELECT 구문 안에서 보조로 사용되는 또 다른 SELECT 구문이다.

SELECT 구문을 효율적으로 작성할 수 있도록 도와준다.

복잡한 SELECT 구문을 작성할 때 필수적으로 많이 사용되는 기법이다.

SELECT 열이름1, 열이름2, 열이름3...

FROM 테이블 이름

WHERE 조건식

(SELECT 열이름1, 열이름2, 열이름3....

FROM 테이블 이름

WHERE 조건식);

서브 쿼리는 논리가 복잡한 SQL 구문에서는 거의 필수로 많이 사용된다.

- 서브쿼리는 () 괄호로 묶어서 사용한다. 메인은 묶지 않는다.
- 메인쿼리와 서브 쿼리를 연결하기 위해 단일행 연산자 또는 다중행 연산자를 사용한다.
- 서브쿼리(괄호안)가 먼저 실행되고 그 다음 메인쿼리가 실행된다.
- 서브 쿼리의 서브쿼리의 서브쿼리..... 형태로 계속 중첩시킬 수 있다.

### 서브 쿼리의 종류

- 단일행 서브 쿼리 - 하나의 행을 검색하는 서브쿼리
- 다중행 서브 쿼리 - 하나 이상의 행을 검색하는 서브 쿼리
- 다중열 서브 쿼리 - 하나 이상의 열을 검색하는 서브쿼리

- 
1. 단일행 서브 쿼리 - 서브쿼리 SELECT 문에서 얻은 한 개의 행의 결과값을 메인 쿼리로 전달하는 서브 쿼리다.
-



## Quiz 8

employees 테이블의 last\_name 이 'De Haan' 인 직원과 salary 가 동일한 직원을 단일행 서브쿼리를 이용하여 검색하여 출력하시오.

### SQL Worksheet

```
1 SELECT a.salary
2 FROM hr.employees a
3 WHERE a.last_name = 'De Haan';
```

SALARY

17000

[Download CSV](#)

### SQL Worksheet

[Clear](#)

[Find](#)

[Actions](#)

[Save](#)

[Run](#)

```
1 SELECT *
2 FROM hr.employees a
3 WHERE salary = (
4     SELECT a.salary
5     FROM hr.employees a
6     WHERE a.last_name = 'De Haan'
7 );
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	-	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	-	100	90

[Download CSV](#)

2 rows selected.

```
SELECT *
FROM hr.employees a
WHERE salary = (
    SELECT a.salary
    FROM hr.employees a
    WHERE a.last_name = 'De Haan'
);
```

employees 테이블의 last\_name 이 'Taylor' 인 직원과 salary 가 동일한 직원에 대한 모든 정보를 단일행 서브쿼리를 이용하여 검색하여 출력하시오.

SQL Worksheet
Clear
Find
Actions
Save
Run

```

1 SELECT *
2 FROM hr.employees a
3 WHERE (a.last_name, a.salary) IN (
4         SELECT a.last_name, a.salary
5         FROM hr.employees a
6         WHERE a.last_name = 'Taylor'
7     );
8
9

```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
176	Jonathon	Taylor	JTAYLOR	011.44.1644.429265	24-MAR-06	SA_REP	8600	.2	149	80
180	Winston	Taylor	WTAYLOR	650.507.9876	24-JAN-06	SH_CLERK	3200	-	120	50

Download CSV  
2 rows selected.

```

SELECT *
FROM hr.employees a
WHERE (a.last_name, a.salary) IN (
        SELECT a.last_name, a.salary
        FROM hr.employees a
        WHERE a.last_name = 'Taylor'
    );

```

## 다중행 서브 쿼리

사용법은 단일행과 동일하다. 다중행 서브쿼리는 하나 이상의 결과를 메인 쿼리에 전달하는 경우에 사용된다.

IN	같은 값이 여러 개	IN(1,2,3)
NOT IN	같은 값이 아닌 여러 개	NOT IN(1,2,3)

EXIST	값이 있으면 True, 없으면 False	
ANY	최소한 하나라도 존재 (or)	ANY(10, 20)
ALL	모두 만족 (and)	ALL(10, 20)

## Quiz 9

employees 테이블에서 department\_id 별로 가장 낮은 salary가 얼마인지 찾아서 그에 해당하는 직원이 누구인지 다중행 서브 쿼리를 이용하여 찾아보자.

### SQL Worksheet

```

1 SELECT a.department_id, MIN(salary)
2 FROM hr.employees a
3 GROUP BY department_id;
4
5

```

DEPARTMENT_ID	MIN(SALARY)
50	2100
40	6500
110	8300
90	17000
30	2500
70	10000
-	7000
10	4400
20	6000
60	4200
100	6900
80	6100

Download CSV  
12 rows selected.

```

SELECT a.department_id, MIN(salary)
FROM hr.employees a
GROUP BY department_id;

```

## SQL Worksheet



Actions ▾



```

1 SELECT *
2 FROM hr.employees a
3 WHERE salary IN(
4         SELECT MIN(salary)
5         FROM hr.employees a
6         GROUP BY department_id
7     )
8 ORDER BY a.salary DESC;
9

```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	-	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	-	100	90
204	Hermann	Baer	HBAER	515.123.8888	07-JUN-02	PR_REP	10000	-	101	70
169	Harrison	Bloom	HBLOOM	011.44.1343.829268	23-MAR-06	SA_REP	10000	.2	148	80
156	Janette	King	JKING	011.44.1345.429268	30-JAN-04	SA_REP	10000	.35	146	80
150	Peter	Tucker	PTUCKER	011.44.1344.129268	30-JAN-05	SA_REP	10000	.3	145	80
206	William	Gietz	WGIEZT	515.123.8181	07-JUN-02	AC_ACCOUNT	8300	-	205	110
161	Sarath	Sewall	SSEWALL	011.44.1345.529268	03-NOV-06	SA_REP	7000	.25	146	80
178	Kimberely	Grant	KGRANT	011.44.1644.429263	24-MAY-07	SA_REP	7000	.15	149	-
155	Oliver	Tuvault	OTUVAULT	011.44.1344.486508	23-NOV-07	SA_REP	7000	.15	145	80
113	Luis	Popp	LPOPP	515.124.4567	07-DEC-07	FI_ACCOUNT	6900	-	108	100
123	Shanta	Vollman	SVOLLMAN	650.123.4234	10-OCT-05	ST_MAN	6500	-	100	50
203	Susan	Mavris	SHAVRIS	515.123.7777	07-JUN-02	HR_REP	6500	-	101	40
173	Sundita	Kumar	SKUMAR	011.44.1343.329268	21-APR-08	SA_REP	6100	.1	148	80
104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-07	IT_PROG	6000	-	103	60
202	Pat	Fay	PFAY	603.123.6666	17-AUG-05	MK_REP	6000	-	201	20
200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-03	AD_ASST	4400	-	101	10
107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-07	IT_PROG	4200	-	103	60
184	Nandita	Sarchand	NSARCHAN	650.509.1876	27-JAN-04	SH_CLERK	4200	-	121	50
131	James	Marlow	JAMRLOW	650.124.7234	16-FEB-05	ST_CLERK	2500	-	121	50
191	Randall	Perkins	RPERKINS	650.505.4876	19-DEC-07	SH_CLERK	2500	-	122	50
140	Joshua	Patel	JPATEL	650.121.1834	06-APR-06	ST_CLERK	2500	-	123	50
119	Karen	Colmenares	KCOLMENA	515.127.4566	10-AUG-07	PU_CLERK	2500	-	114	30

```

SELECT *
FROM hr.employees a
WHERE salary IN
    (SELECT MIN(salary)
     FROM hr.employees a
     GROUP BY department_id
    )
ORDER BY a.salary DESC;

```

다중 열 서브쿼리

메인 쿼리와 서브 쿼리를 비교하는 WHERE 조건식에서 비교되는 열이 여러개일 경우 사용되는 서브 쿼리이다.

```
SELECT 열이름1, 열이름2, 열이름3...  
FROM 테이블 이름  
WHERE (열이름1, 열이름2, 열이름3...) IN  
      (SELECT 열이름1, 열이름2, 열이름3....  
        FROM 테이블 이름  
        WHERE 조건식);
```

---

#### Quiz 10

employees 테이블에서 job\_id 별로 가장 높은 salary가 얼마인지 찾아보고 찾아낸 job\_id별 salary에 해당하는 직원이 누구인지 다중 열 서브쿼리를 사용하여 찾아보자.

```

1 SELECT *
2 FROM hr.employees a
3 WHERE (a.job_id, a.salary) IN(
4         SELECT job_id, MAX(salary)
5         FROM hr.employees a
6         GROUP BY a.job_id
7     )
8 ORDER BY a.salary DESC;
9

```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000	-	-	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	-	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	-	100	90
145	John	Russell	JRUSSEL	011.44.1344.429268	01-OCT-04	SA_MAN	14000	.4	100	80
201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-04	MK_MAN	13000	-	100	20
108	Nancy	Greenberg	NGREENBE	515.124.4569	17-AUG-02	FI_MGR	12008	-	101	100
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-02	AC_MGR	12008	-	101	110
168	Lisa	Ozer	LOZER	011.44.1343.929268	11-MAR-05	SA_REP	11500	.25	148	80
114	Den	Raphaely	DRAPHEAL	515.127.4561	07-DEC-02	PU_MAN	11000	-	100	30
204	Hermann	Baer	HBAER	515.123.8888	07-JUN-02	PR_REP	10000	-	101	70
103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000	-	102	60
109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI_ACCOUNT	9000	-	108	100
206	William	Gietz	WGIEZT	515.123.8181	07-JUN-02	AC_ACCOUNT	8300	-	205	110
121	Adam	Fripp	AFRIPP	650.123.2234	10-APR-05	ST_MAN	8200	-	100	50
203	Susan	Mavris	SMAVRIS	515.123.7777	07-JUN-02	HR_REP	6500	-	101	40
202	Pat	Fay	PFAY	603.123.6666	17-AUG-05	MK_REP	6000	-	201	20
200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-03	AD_ASST	4400	-	101	10
184	Nandita	Sarchand	NSARCHAN	650.509.1876	27-JAN-04	SH_CLERK	4200	-	121	50
137	Renske	Ladwig	RLADWIG	650.121.1234	14-JUL-03	ST_CLERK	3600	-	123	50
115	Alexander	Khoo	AKHOO	515.127.4562	18-MAY-03	PU_CLERK	3100	-	114	30

Download CSV

20 rows selected.

```

SELECT *
FROM hr.employees a
WHERE (a.job_id, a.salary) IN(
        SELECT job_id, MAX(salary)
        FROM hr.employees a
        GROUP BY a.job_id
    )
ORDER BY a.salary DESC;

```

