실기과제#4

1876217 컴퓨터공학과 안서연

목차

1. INNER JOIN
2. SELF JOIN
3. CROSS JOIN
4. OUTER JOIN
5. SUBQUERY
   -SINGLE SUBQUERY
   -MULTIPLE SUBQUERY

1. INNER JOIN

1.1 INNER NATURAL J0IN

```
SELECT COUNTRY_ID,COUNTRY_NAME,REGION_ID,REGION_NAME
FROM COUNTRIES NATURAL JOIN REGIONS;
```

Natural join을 사용하여 countries 참조테이블에서 country_id,country_name 열과 regions 참조테이블의 region_id,region_name열을 합쳤다. 총 25행이 출력되었다.

<출력결과>

	COUNTRY_ID	COUNTRY_NAME	REGION_ID	REGION_NAME
1	NL	Netherlands	1	Europe
2	FR	France	1	Europe
3	UK	United Kingdom	1	Europe
4	DK	Denmark	1	Europe
5	BE	Belgium	1	Europe
6	CH	Switzerland	1	Europe
7	IT	Italy	1	Europe
8	DE	Germany	1	Europe
9	US	United States of America	2	Americas
10	CA	Canada	2	Americas
11	MX	Mexico	2	Americas
12	BR	Brazil	2	Americas
13	AR	Argentina	2	Americas
14	ML	Malaysia	3	Asia
15	JP	Japan	3	Asia
16	IN	India	3	Asia
17	CN	China	3	Asia
18	AU	Australia	3	Asia
19	SG	Singapore	3	Asia
20	IL	Israel	4	Middle East and Africa
21	ZM	Zambia	4	Middle East and Africa
22	EG	Egypt	4	Middle East and Africa
23	ZW	Zimbabwe	4	Middle East and Africa
24	NG	Nigeria	4	Middle East and Africa
25	KW	Kuwait	4	Middle East and Africa

*USING을 이용한 JOIN

```
SELECT JOB_TITLE,START_DATE,END_DATE,JOB_ID
FROM JOBS JOIN JOB_HISTORY USING(JOB_ID);
```

Using을 이용하여 jobs 참조테이블과 job_history 테이블에 공통적으로 존재하는 job_id 열에 대하여 jobs참조테이블의 job_title,start_date,end_date,job_id를 출력하였다. 총 10행이 출력되었다.

<출력결과>

	JOB_TITLE	START_DATE	END_DATE	JOB_ID
1	Public Accountant	97/09/21	01/10/27	AC_ACCOUNT
2	Public Accountant	02/07/01	06/12/31	AC_ACCOUNT
3	Accounting Manager	01/10/28	05/03/15	AC_MGR
4	Administration Assistant	95/09/17	01/06/17	AD_ASST
5	Programmer	01/01/13	06/07/24	IT_PROG
6	Marketing Representative	04/02/17	07/12/19	MK_REP
7	Sales Manager	07/01/01	07/12/31	SA_MAN
8	Sales Representative	06/03/24	06/12/31	SA_REP
9	Stock Clerk	07/01/01	07/12/31	ST_CLERK
10	Stock Clerk	06/03/24	07/12/31	ST_CLERK

1. INNER JOIN

1.2 INNER EQUI JOIN

두 테이블에 동일한 열 이름이 존재하는 경우

```
SELECT C.COUNTRY_NAME,L.CITY,C.COUNTRY_ID,L.COUNTRY_ID
FROM COUNTRIES C
JOIN LOCATIONS L
ON C.COUNTRY_ID=L.COUNTRY_ID
```

Countries 참조테이블을 C라는 가명으로, locations 테이블을 L이라는 가명으로 하여 두 테이블의 country_id 가 같은 것을 기준으로 countries 테이블의 country_name,city,country_id를, locations 테이블의 country_id 를 출력하였다. 총 23행이 출력되었다.

<출력결과>

	COUNTRY_NAME	CITY	COUNTRY_ID	COUNTRY_ID_1
1	Australia	Sydney	AU	AU
2	Brazil	Sao Paulo	BR	BR
3	Canada	Toronto	CA	CA
4	Canada	Whitehorse	CA	CA
5	Switzerland	Geneva	CH	CH
6	Switzerland	Bern	CH	CH
7	China	Beijing	CN	CN
8	Germany	Munich	DE	DE
9	India	Bombay	IN	IN
10	Italy	Roma	IT	IT
11	Italy	Venice	IT	IT
12	Japan	Tokyo	JP	JP
13	Japan	Hiroshima	JP	JP
14	Mexico	Mexico City	MX	MX
15	Netherlands	Utrecht	NL	NL
16	Singapore	Singapore	SG	SG
17	United Kingdom	London	UK	UK
18	United Kingdom	Oxford	UK	UK
19	United Kingdom	Stretford	UK	UK
20	United States of America	Southlake	US	US
21	United States of America	South San Francisco	US	US
22	United States of America	South Brunswick	US	US
23	United States of America	Seattle	US	US

*3-WAY JOIN

```
SELECT FIRST_NAME,LAST_NAME,E.JOB_ID,E.EMPLOYEE_ID
FROM EMPLOYEES E
JOIN JOBS J
ON E.JOB_ID=J.JOB_ID
JOIN JOB_HISTORY H
ON E.EMPLOYEE_ID=H.EMPLOYEE_ID;
```

3가지 테이블에서 겹치는것을 확인하는 join으로,각각 employees 테이블은 E, jobs 테이블은 J, job_history 테이블은 H로 하여 E와 J의 job_id가 같으면서 E의 employee_id와 H의 employee_id가 같은 employees 테이블 행에서 first_name,last_name,job_id,employee_id를 출력하였다. 총 10행이 출력되었다.

<출력결과>

	FIRST_NAME	LAST_NAME	JOB_ID	EMPLOYEE_ID
1	Neena	Kochhar	AD_VP	101
2	Neena	Kochhar	AD_VP	101
3	Lex	De Haan	AD_VP	102
4	Den	Raphaely	PU_MAN	114
5	Payam	Kaufling	ST_MAN	122
6	Jonathon	Taylor	SA_REP	176
7	Jonathon	Taylor	SA_REP	176
8	Jennifer	Whalen	AD_ASST	200
9	Jennifer	Whalen	AD_ASST	200
10	Michael	Hartstein	MK_MAN	201

2. SELF-JOIN

```
SELECT L.CITY,C.STATE_PROVINCE,L.POSTAL_CODE,C.POSTAL_CODE
FROM LOCATIONS L
JOIN LOCATIONS C
ON(L.CITY=C.STATE_PROVINCE);
```

Self-join은 자기 자신의 테이블을 대상으로 join을 진행하는것이다. Locations 참조테이블에 대하여 같은 테이블이지만 L,C라는 가명을 두어 해당 참조 테이블 내에서 city와 state_province 명이 같은것을 만족하는 locations 테이블의 city,state_province,과 값이 같은지 확인하기위하여 L의 postal_code와 C의 postal_code를 출력하였다. 결과 postal_code는 서로 같아고, 총 3행이 출력되어다. 도시명과 state province 명이 같은 곳이 3곳임을 알수 있다.

<출력결과>

	CITY	STATE_PROVINCE	POSTAL_CODE	POSTAL_CODE_1
1	Oxford	Oxford	OX9 9ZB	OX9 9ZB
2	Sao Paulo	Sao Paulo	01307-002	01307-002
3	Utrecht	Utrecht	3029SK	3029SK

3. CROSS JOIN

```
SELECT COUNTRY_ID,REGION_NAME
FROM COUNTRIES CROSS JOIN REGIONS
```

<출력결과>

Cross join은 한테이블의 모든 행과 다른 테이블의 모든 곱집합, catesian product를 출력하는 join이다. Countries 참조테이블에서 regions테이블에 대하여 Countries 테이블의 country_id와 regions 테이블의 region_name의 곱집합을 형성하였다. Countries 테이블은 총 25행이고 regions는 총 4행이다. 25*4의 곱집합이 형성되어 출력결과와 같이 100행이 출력되었다.

	COUNTRY_ID	REGION_NAME
1	AR	Europe
2	AU	Europe
3	BE	Europe
4	BR	Europe
5	CA	Europe
6	CH	Europe
7	CN	Europe
8	DE	Europe
9	DK	Europe
10	EG	Europe
11	FR	Europe
12	IL	Europe
13	IN	Europe
14	IT	Europe
15	JP	Europe
16	KW	Europe
17	ML	Europe
18	MX	Europe
19	NG	Europe
20	NL	Europe

...

83	DE	Middle East a...
84	DK	Middle East a...
85	EG	Middle East a...
86	FR	Middle East a...
87	IL	Middle East a...
88	IN	Middle East a...
89	IT	Middle East a...
90	JP	Middle East a...
91	KW	Middle East a...
92	ML	Middle East a...
93	MX	Middle East a...
94	NG	Middle East a...
95	NL	Middle East a...
96	SG	Middle East a...
97	UK	Middle East a...
98	US	Middle East a...
99	ZM	Middle East a...
100	ZW	Middle East a...

4. OUTER JOIN

4.1 LEFT OUTER JOIN

```
SELECT C.COUNTRY_ID,C.REGION_ID,R.REGION_NAME
FROM COUNTRIES C
LEFT OUTER JOIN REGIONS R
ON(C.REGION_ID=R.REGION_ID)
```

Left outer join은 왼쪽 테이블을 기준으로 다른 테이블의 내용을 출력한다. C라는 가명을 가진 countries 참조테이블과 R이라는 가명을 가진 Region 테이블에서 서로 region_id가 같은 조건에서 C의 country_id, region_id,R의 region_name을 출력하였다. 총 25행이 출력되었다.

<출력결과>

	COUNTRY_ID	REGION_ID	REGION_NAME
1	UK	1	Europe
2	NL	1	Europe
3	IT	1	Europe
4	FR	1	Europe
5	DK	1	Europe
6	DE	1	Europe
7	CH	1	Europe
8	BE	1	Europe
9	US	2	Americas
10	MX	2	Americas
11	CA	2	Americas
12	BR	2	Americas
13	AR	2	Americas
14	SG	3	Asia
15	ML	3	Asia
16	JP	3	Asia
17	IN	3	Asia
18	CN	3	Asia
19	AU	3	Asia
20	ZW	4	Middle East and Africa
21	ZM	4	Middle East and Africa
22	NG	4	Middle East and Africa
23	KW	4	Middle East and Africa
24	IL	4	Middle East and Africa
25	EG	4	Middle East and Africa

4.2 RIGHT OUTER JOIN

```
SELECT D.DEPARTMENT_NAME,D.LOCATION_ID,L.CITY
FROM DEPARTMENTS D
RIGHT OUTER JOIN LOCATIONS L
ON(D.LOCATION_ID=L.LOCATION_ID);
```

Right outer join은 오른쪽 테이블을 기준으로 다른 테이블의 내용을 출력한다. D라는 가명을 가진 departments 참조테이블과 L이라는 가명을 가진 location 테이블에서 서로 location_id가 같은 조건에서 D의 department_name,location_id, L의 city를 출력하였다. 총 43행이 출력되었다.

<출력결과>

	DEPARTMENT_NAME	LOCATION_ID	CITY
1	Administration	1700	Seattle
2	Marketing	1800	Toronto
3	Purchasing	1700	Seattle
4	Human Resources	2400	London
5	Shipping	1500	South San Francisco
6	IT	1400	Southlake
7	Public Relations	2700	Munich
8	Sales	2500	Oxford
9	Executive	1700	Seattle
10	Finance	1700	Seattle
11	Accounting	1700	Seattle
12	Treasury	1700	Seattle
13	Corporate Tax	1700	Seattle
14	Control And Credit	1700	Seattle
15	Shareholder Services	1700	Seattle
16	Benefits	1700	Seattle

...

35	(null)	(null)	Geneva
36	(null)	(null)	Sao Paulo
37	(null)	(null)	Venice
38	(null)	(null)	Whitehorse
39	(null)	(null)	Singapore
40	(null)	(null)	Hiroshima
41	(null)	(null)	Stretford
42	(null)	(null)	Roma
43	(null)	(null)	Mexico City

4. OUTER JOIN

4.3 FULL OUTER JOIN

```
SELECT E.FIRST_NAME,E.LAST_NAME,E.MANAGER_ID,D.DEPARTMENT_NAME
FROM EMPLOYEES E FULL OUTER JOIN DEPARTMENTS D
ON (E.MANAGER_ID=D.MANAGER_ID);
```

Full outer join은 두 테이블의 합집합을 출력한다. Employees 참조테이블은 E라는 가명으로, departments 테이블은 D라는 가명으로 하여 두 테이블에서 manager_id가 같은 조건을 만족하는 E의 first_name,last_name,manager_id와 department_name의 합집합을 출력하였따. 총 126행이 출력되었다.

<출력결과>

	FIRST_NAME	LAST_NAME	MANAGER_ID	DEPARTMENT_NAME
1	Steven	King	(null)	(null)
2	Neena	Kochhar	100	Executive
3	Lex	De Haan	100	Executive
4	Alexander	Hunold	102	(null)
5	Bruce	Ernst	103	IT
6	David	Austin	103	IT
7	Valli	Pataballa	103	IT
8	Diana	Lorentz	103	IT
9	Nancy	Greenberg	101	(null)
10	Daniel	Faviet	108	Finance
11	John	Chen	108	Finance
12	Ismael	Sciarra	108	Finance
13	Jose Manuel	Urman	108	Finance
14	Luis	Popp	108	Finance
15	Den	Raphaely	100	Executive
16	Alexander	Khoo	114	Purchasing
17	Shelli	Baida	114	Purchasing
18	Sigal	Tobias	114	Purchasing

...

112	(null)	(null)	(null)	IT Helpdesk
113	(null)	(null)	(null)	NOC
114	(null)	(null)	(null)	IT Support
115	(null)	(null)	(null)	Operations
116	(null)	(null)	(null)	Contracting
117	(null)	(null)	(null)	Construction
118	(null)	(null)	(null)	Manufacturing
119	(null)	(null)	(null)	Benefits
120	(null)	(null)	(null)	Shareholder Services
121	(null)	(null)	(null)	Control And Credit
122	(null)	(null)	(null)	Corporate Tax
123	(null)	(null)	(null)	Treasury
124	(null)	(null)	(null)	Public Relations
125	(null)	(null)	(null)	Human Resources
126	(null)	(null)	(null)	Administration

5. SUBQUERY

5.1 SINGLE-ROW SUBQUERY

```
SELECT DEPARTMENT_NAME,LOCATION_ID
FROM DEPARTMENTS
WHERE LOCATION_ID >(SELECT LOCATION_ID
                    FROM DEPARTMENTS
                    WHERE DEPARTMENT_ID=10);
```

<출력결과>

	DEPARTMENT_NAME	LOCATION_ID
1	Marketing	1800
2	Human Resources	2400
3	Sales	2500
4	Public Relations	2700

Subquery의 70%인 where뒤에 쿼리가 붙는
서브쿼리는 main 쿼리 전에 실행이된다. Departments 참조테이블에서 (서브쿼리)departments 테이블에서 department_id가 10인 행의 location_id 보다 값이 큰 location_id를 가진 행에 대하여 department_name,location_id를 출력하였다. 총 4행이 출력되었다.

```
SELECT JOB_ID,MAX_SALARY
FROM JOBS
WHERE  MAX_SALARY>(SELECT MAX_SALARY
                   FROM JOBS
                   WHERE JOB_ID='IT_PROG')
AND    MIN_SALARY>(SELECT MIN_SALARY
                   FROM JOBS
                   WHERE JOB_ID='IT_PROG');
```

<출력결과>

	JOB_ID	MAX_SALARY
1	AD_PRES	40000
2	AD_VP	30000
3	FI_MGR	16000
4	AC_MGR	16000
5	SA_MAN	20080
6	SA_REP	12008
7	PU_MAN	15000
8	MK_MAN	15000
9	PR_REP	10500

단일행에 대한 조건이 2개인 경우이다. Jobs 참조테이블에서 (서브쿼리)jobs 테이블 내에서 job_id가 IT_PROG인 행의 max_salary 보다 max_salary가 많고, (서브쿼리) jobs테이블 내에서 job_id가 IT_PROG인 행의 min_salary 보다 MIN_SALARY가 큰 행에 대하여 job_id,max_salary를 출력하였다. 즉 IT_PROG보다 최저/최대 급여가 높은 직업을 찾는것이다. 총 9행이 출력되었다.

5. SUBQUERY

5.1 SINGLE-ROW SUBQUERY

*그룹함수, HAVING 사용할 경우

```
SELECT DEPARTMENT_ID,MIN(HIRE_DATE)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID
HAVING MIN(HIRE_DATE)< (SELECT MIN(HIRE_DATE)
                        FROM EMPLOYEES
                        WHERE DEPARTMENT_ID=20);
```

<출력결과>

	DEPARTMENT_ID	MIN(HIRE_DATE)
1	100	02/08/16
2	30	02/12/07
3	90	01/01/13
4	70	02/06/07
5	110	02/06/07
6	50	03/05/01
7	80	04/01/30
8	40	02/06/07
9	10	03/09/17

Employees참조테이블에서 department_id 단위로 그룹을 묶고, (서브쿼리)employees 테이블에서 department_id가 20인 조건에서 hire_date값의 최솟값(가장 오래 근무한사람) 에 대하여 그룹중에 hire_date값의 최솟값이 이보다 작은 그룹의 department_id와 hire_date를 출력하였다. 즉 department_id가 20 인 곳에서 가장 오래 근무를 시작한 사람보다 더 일찍 근무한 사람을 찾는것이다. 총 9행이 출력되었다.

5. SUBQUERY

5.2 MUTIPLE-ROW SUBQUERY

```
SELECT EMPLOYEE_ID,LAST_NAME,JOB_ID,COMMISSION_PCT
FROM EMPLOYEES
WHERE COMMISSION_PCT >ALL
                        (SELECT COMMISSION_PCT
                         FROM EMPLOYEES
                         WHERE JOB_ID='SA_REP')
                         AND JOB_ID<>'SA_REP' ;
```

<출력결과>

	EMPLOYEE_ID	LAST_NAME	JOB_ID	COMMISSION_PCT
1	145	Russell	SA_MAN	0.4

다중 행 서브쿼리는 두개 이상의 행을 반환하면서 조건을 검색하는 쿼리이다. Employees 테이블에서 (서브쿼리)employees 테이블에서 job_id가 SA_REP인 행의 모든 commission_pct보다 큰 commission_pct를 가지고, job_id가 SA_REP가 아닌 행의 employee_id,last_name,job_id,commisision_pct를 출력하였다. 총 1가지의 행이 출력되었다.

5.3 MUTIPLE-COLUMN SUBQUERY

```
SELECT FIRST_NAME,JOB_ID,HIRE_DATE
FROM EMPLOYEES
WHERE(HIRE_dATE,JOB_ID) IN
     (SELECT MIN(HIRE_DATE),JOB_ID
      FROM EMPLOYEES
      GROUP BY JOB_ID)
ORDER BY JOB_ID;
```

다중 행 서브쿼리는 두개 이상의 열을 반환하면서 조건을 검색하는 쿼리이다.
Employees 테이블에서 (서브쿼리) job_id로 그룹화할때 각 그룹의 hire_date의 최솟값 min(hire_date)와 job_id 를 반환하는데, employees 테이블에서 job_id로 그룹화 했을 때 (hire_date,job_id)가 이 열들에 만족되는 것이 있으면 first_name,job_id,hire_date를 job_id에 대하여 정렬하여 출력하였다. 총 19행이 출력되었다.

<출력결과>

	FIRST_NAME	JOB_ID	HIRE_DATE
1	William	AC_ACCOUNT	02/06/07
2	Shelley	AC_MGR	02/06/07
3	Jennifer	AD_ASST	03/09/17
4	Steven	AD_PRES	03/06/17
5	Lex	AD_VP	01/01/13
6	Daniel	FI_ACCOUNT	02/08/16
7	Nancy	FI_MGR	02/08/17
8	Susan	HR_REP	02/06/07
9	David	IT_PROG	05/06/25
10	Michael	MK_MAN	04/02/17
11	Pat	MK_REP	05/08/17
12	Hermann	PR_REP	02/06/07
13	Alexander	PU_CLERK	03/05/18
14	Den	PU_MAN	02/12/07
15	John	SA_MAN	04/10/01
16	Janette	SA_REP	04/01/30
17	Nandita	SH_CLERK	04/01/27
18	Renske	ST_CLERK	03/07/14
19	Payam	ST_MAN	03/05/01